COMPUTERS

Microcomputers in Plain English

for New Zealanders

Brian D. Strong

MICRO

Microcomputers in Plain English

for New Zealanders

Microcomputers in Plain English
for New Zealanders

Brian D. Strong

Illustrations by Mark Raffety

First published 1983

A. H. & A. W. REED LTD

68–74 Kingsford-Smith Street, Wellington 3

also

7 Kirk Street, Auckland 2
85 Thackeray Street, Christchurch 2

Copyright ©1983 Brian D. Strong

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission of the publishers.

ISBN 0 589 01466 8

Typesetting by Quickset, Christchurch, New Zealand.
Printed by Wing King Tong Co Ltd, Hong Kong

Contents

Introduction		vii
Chapter 1	How it all happened	1
Chapter 2	Translating computerspeak	11
Chapter 3	How computers work	20
Chapter 4	The add-ons and extras	29
Chapter 5	What to do with it now you've got it	37
Chapter 6	Looking at languages	47
Chapter 7	About programming	53
Chapter 8	Programs look like this	71
Chapter 9	Choosing your micro and software	82
Chapter 10	Looking ahead – some personal views	92
Chapter 11	So you want to know more	101
	Glossary	105
	Index	110

Author's note

Because microcomputers are usually programmed using American spellings, any words related to computer functions and terminology have been given such spellings in this book (for example, catalog, program, disk). A full glossary may be found on page 105.

All commands used in computer programs are given in uppercase letters (for example, RUN, LIST, PRINT). The same applies to the names of all computer languages (for example, BASIC, COBOL, FORTRAN).

Introduction

If you are already familiar with microcomputers, or have had some training in computer science, this book is not for you. Those who feel they are being swallowed up by new technology and those who know nothing about computers, or perhaps just want to gain a working knowledge of them without being blinded by obscure words and technical jargon, may read on. By the time you have finished you will be ready to take on other more detailed books on the subject.

My aim is to help you to understand not only what makes up a microcomputer but also the functions of all the bits and pieces involved and the general principles of operation. We will also look at both the various uses of microcomputers and how to go about selecting one to suit your requirements. Then you will be ready for a conducted tour through some computer programs.

As when driving a car, it is not necessary to understand what happens under the lid in order to operate a machine successfully, but it helps to have a general understanding of what is happening.

Always remember that a microcomputer is not a magic item presented in fancy wrapping. It is a valuable aid for many purposes, but it is also a brainless collection of electrical components and will do only what it is told. If you turn off the switch it will be as useless as a radio with flat batteries.

This book has been written from my own experiences in unravelling the mysterious world of computing. It is the result of both discussions with many people who found themselves suddenly surrounded by new technology they were unable to understand, and encounters with thousands of people who visited a microcomputer exhibition to see what computers looked like and with the intention of asking questions. After three days, I lost my voice answering them. This book is derived from the ques-

tions people have asked and the explanations are those that were readily understood.

In the interests of clarity I have taken a few liberties from time to time, but they will not lead you astray. A final word of warning: microcomputers are terribly addictive!

Now, let's get going. . . .

Brian Strong

Chapter 1

How it all happened

From the moment of birth we depend on information. We gather it through the senses of sight, sound, smell and touch. Once information is accumulated it is then processed in our brains and from the results we draw various conclusions, by comparing the facts we have with each other or with past experience. Each new set of facts or experiences is filed away in our minds for future use. Thus we each accumulate our own information or "data" banks.

Over the centuries the amount of information available to us has increased to the point where it is impossible to store everything in our heads. We now use computers, which – despite their mystique – are merely machines capable of storing vast amounts of information and able to perform various calculations or draw conclusions from the information they contain. Their advantage is that they are able to do this at incredible speeds.

To appreciate how far we have come in so short a time, it is worthwhile to look back and see where we have been.

Computing is not new. A simple definition of computing is "to determine an amount or number", which is as old as human history.

Taking an imaginative trip back in time, it is fairly safe to assume that this began when primitive peoples discovered that fingers and toes could be used to represent the passage of time or perhaps the number of beasts killed for the group.

Imagine that rivalry between hunters in a group led to tall stories about "the one that got away" or the number of beasts killed during a certain period. As the tales grew it was found there weren't enough fingers and toes to support claims to fame, so the hunters started placing rocks in a pile, with each rock representing a kill – or perhaps they made marks on the walls of a communal cave. Whatever method was used, this was the start of record keeping or the accumulation of information.

Because these early peoples usually led nomadic lifestyles all such records would have been committed to memory or carried with them as markings, possibly on animal hides. After arriving at new locations their records could be reproduced, either for themselves or for others.

As the years passed trade between groups developed, as did the need for more permanent records. The earliest known form of written record was kept by the Sumerians during 3000–2600 B.C. and consisted of symbols scratched into tablets of wet clay which were then baked dry in the sun.

Clay tablets continued to be used by Mediterranean cultures until the Egyptians invented papyrus. This early form of paper was made from a water plant which grew on the banks of the Nile.

The next step forward occurred about 1 A.D. when the Greeks and Romans replaced papyrus with parchment made from animal skins. It wasn't until late in the fourteenth century that parchment was replaced in the western world by a new material called paper. History also reveals that paper was used by the Chinese as early

as 2 A.D. With paper came the quill pen, which gave way to metal pens in the 1800s and fountain pens near the year 1900.

Despite the emphasis history places on pens, the humble pencil is sometimes forgotten. The forerunners of the pencil – graphite rods – were used as standard writing instruments in the sixteenth century. About 1680–90 an unknown inventor packed a graphite rod inside an outer layer of wood and gave us the pencil as we know it today.

After the invention of pens, pencils and paper as devices for recording information the next significant development was the typewriter. The first patent for a typewriter was filed in 1714 by a Mr Henry Milne, an Englishman, but it was not until the 1800s that a really workable machine was produced by an American, Christopher Sholes of Milwaukee. In 1873 he signed a contract with E. Remington & Sons for its manufacture, but it took another four or five years before the typewriter began to be accepted as superior in speed and legibility to the pen.

The typewriter paved the way for the development of office equipment such as account printing machines and addressographs and is the model for many of the input and output devices found on the computer today.

Hand-in-hand with the development of various recording devices was the refinement of the written word. Before the 1400s all books were beyond the reach of the ordinary person. Usually religious volumes slavishly reproduced by hand in monasteries, these huge and highly decorated texts were fixed by chains to desks in churches to be marvelled over by illiterate populations.

Books were not produced in quantity until 1450, when Johann Gutenberg published Bibles printed with movable metal type on a printing press. This is what really began the revolution in communications which brought knowledge to everyone who could read.

Let us now return to that pile of rocks and trace the development of computing.

Finger counting, piles of rocks and the more portable version, a bag of pebbles, finally evolved into the first "counting machine" – the abacus. Believed to have originated in China about 2600 B.C., the simple abacus filled the role of today's pocket calculator and is still in use in many Eastern countries.

The abacus consists of wires or rods placed across a rectangular frame. Mounted on these wires or rods are wooden beads. The operator carries out calculations by sliding these beads into various positions according to a strict set of rules, the position of a particular bead on a column denoting its value.

Skilled abacus operators can calculate very quickly, and in speed contests that used to be held between the abacus and electric adding machines the 5,000-year-old marvel often came out on top.

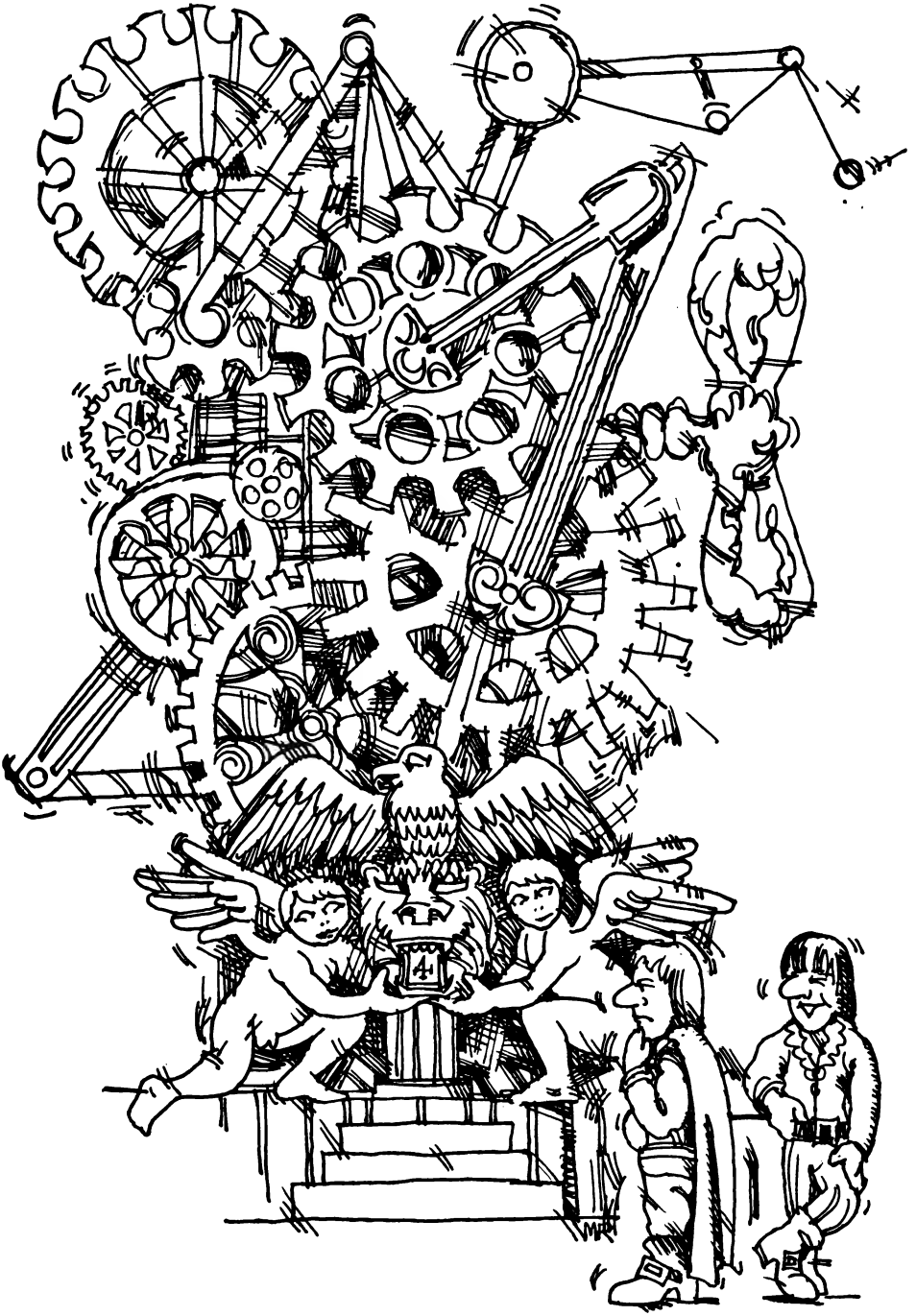
As time progressed attempts were made to add up numbers by mechanical means, and in 1617 a method of calculating by machine was announced by John Napier, a distinguished Scottish mathematician. John Napier was also the inventor of logarithms, another form of calculation currently passing into history. (Logarithms, for those who have never experienced them, are complex tables of numbers used as aids to simplify multiplication and division.)

Napier's calculating machine arose from the idea of laying out multiplication tables on cylinders which could be mounted in a frame and rotated. Each cylinder or rod carried the digits 1 to 9 with their multiples below. It was operated by turning a handle underneath the frame. Numbers showing up on the face were then added or subtracted. Because the original rods were made of ivory these calculating machines became known as "Napier's Bones".

A major drawback from this invention was its size. This may have provided the spur for Edmund Gunter, who must have needed something capable of being carried around in his pocket or bag. Gunter took two of Napier's logarithmic scales and reproduced them on two strips of wood which could slide alongside each other. This was the forerunner of the slide rule, an instrument that was used for three centuries until pocket calculators came on the scene in the mid 1970s.

The next development came from a 19-year-old youth who found himself working in his father's tax office in Rouen, France. Tired of performing the same calculations day after day, Blaise Pascal thought there must be an easier way of carrying out his endless task. Pascal, who went on to make an impact on seventeenth-century science and philosophy, applied himself to the problem and in 1642 invented the first machine that could truly perform arithmetical functions.

The machine was comprised of a series of axles and cogs which interlocked through gears and could handle as many as eight columns of figures. The whole mechanism was housed in a box about the size of a shoebox. On the outside was a series of wheels, and above them a series of corresponding windows. Operation was simple. One merely "dialled" the required figures on the outside wheels according to the value of the number and the column in which it was situated. As the cogs and gears moved



"We're working on the pocket version."

around inside, the resulting answer was displayed through the windows on the front of the box. It caused quite a stir. Here was a machine that could add, subtract, multiply by a series of additions and divide by a series of subtractions.

In spite of these attributes and the sensation Pascal's machine caused, it was an economic flop. Businessmen of the day preferred the cheap clerical labour available and considered Pascal's machine, although amazing, cumbersome to operate and capable of only a portion of the work carried out by their clerks.

Despite setbacks caused by people's attitudes, technology has a habit of moving forward. Shortly after Pascal assembled his first machine Gottfried Leibnitz was born, and he was to set the scene for the next advance.

After examining one of Pascal's machines Leibnitz came to the conclusion that in order to be truly useful the mechanics of a calculator would have to operate on a different principle. Leibnitz replaced the multi-toothed cog by a wheel fitted with nine teeth of different lengths. This speeded up operations. So efficient was this principle that it was maintained in mechanical adding machines and calculators into the 1960s. Unfortunately, Leibnitz proved to be like many men with great minds. Having solved the problem, he became bored with it and transferred his attention to other things.

The next significant step occurred 200 years later when English inventor and mathematician Charles Babbage came up with what was first called the "Difference Engine" and then the "Analytical Engine".

Babbage was looking for a way to relieve routine and time-wasting methods of handling and calculating numerical data. He was convinced he could assemble a machine to do the job. Unfortunately, his idea was ahead of the technical capabilities of the day. His "Difference Engine" was made up of hundreds of rods, wheels, cogs, gears and cams which needed to be manufactured to the most critical tolerances. Despite great care being taken, the demands were greater than Babbage's mechanics could match and, after eating up a vast amount of money, the whole scheme slowly came to a halt.

Undismayed, Babbage then turned his attention to his "Analytical Engine". This was to be designed in such a way that not only would it carry out a variety of functions but it could also be told each time which method to use. This was the first case of research on the possibility of a programmable computer.

For the method of passing information into the machine and telling it what to do Babbage looked into the invention of French

weaver Joseph-Marie Jacquard. Jacquard had invented a loom that used a punched hole in a card to control the weaving process.

Paying the penalty of being born before his time, Babbage gradually lost support for his idea and was unable to raise enough finance to get his machine built. What is now regarded as a brilliant concept slipped into obscurity. Charles Babbage died in 1871 at the age of 80 years. His writings were rediscovered in 1937 and a model of his machine can be found in the British Museum.

This brings us back to 1868 and Remington & Sons and their typewriter. The electric typewriter followed in 1920, providing the basis for bookkeeping machines, addressographs, teletypes, composing machines and data-processing machines.

Jacquard's invention of the punched card for the weaving industry didn't pass unnoticed by others. In 1866 an American statistician, Herman Hollerith, came to the conclusion that a machine could be built to sense such holes. The holes, in turn, could be used to represent numbers. His theory was that quantities of data could be reproduced as punched cards which could be fed through a machine. The machine would then perform calculations on this information and give answers in a fraction of the time normally taken by manual methods.

Hollerith was employed at the United States Census Bureau. After a census had been taken all the cards had to be sorted and resorted time and time again until all the data had been obtained. He thought punched cards would be the answer and his experiments with various electromagnetic devices proved successful.

Data was entered on the cards by means of a hand punch. When a card had to be read it was positioned over rows of mercury-filled cups. Corresponding rows of metal pins were then lowered onto the card. Where there was a hole in the card, the rod went through and rested in the cup of mercury underneath. This caused an electrical connection and one unit was recorded on a dial. For each row on the card there was a corresponding dial and, as the cards were read, the totals on the dials accumulated.

As a result of this invention the 1890 census was completed in just one-third of the time normally taken when using handwritten tally sheets. Thus began the era of the "do not fold, spindle or mutilate" punch card, which forms the basis of data processing on many of our large computer systems.

Hollerith went on to found the Tabulating Machine Company in Washington D.C. In time this became the International Business Machine Corporation and is now known simply as IBM.

In the 1940s two events took place that have had far-reaching effects both on technology and social customs. The first occurred in 1942 with the invention of magnetic tape, which paved the way for the cassette recorder, a multi-million dollar recording industry and a quick and efficient method of storing information. The second was in 1943 when two large computers, Harvard Mark 1 in the U.S.A. and COLOSSUS in England, were switched on for the first time.

Howard Aiken of Harvard University, in association with IBM, started work on Harvard Mark 1 in 1939. Using standard business-machine components and electrical relays, the engineers created a veritable monster of a machine measuring more than 15 metres long, 2.4 metres high and containing about 1 million components. Data input was carried out by holes in punched cards, and output was reproduced by either punched cards or an electric typewriter.

The British machine, COLOSSUS, has been described as looking as if it had been thrown together in a hurry. This description is probably true as COLOSSUS was developed under a shroud of wartime secrecy and little was known of its existence until after World War II.

The biggest technological advance with COLOSSUS was that computing functions were not carried out mechanically but by radio valves (vacuum tubes) – 2,000 of them. Once the first model had been built and was running successfully, nine others were constructed and in operation before the end of the war. These were the world's first electronic digital computers, but they were capable of only one job – cracking German wartime codes. The Germans were confident that their Enigma machines could produce unbreakable codes, but with the help of COLOSSUS the British were finally able to decipher any messages that fell into their hands.

In the U.S.A. at the University of Pennsylvania, also under a veil of secrecy, John Mauchly and Prosper Eckert were developing the machine that is usually recognised as the world's first all-electronic all-purpose computer. This was the Electronic Numerical Integrator and Calculator or, as it is more comfortably known, ENIAC.

Mauchly and Eckert also utilised vacuum tubes for their secret project, but they were unaware of the work being done in England. Two-and-a-half years after construction had commenced ENIAC was finally switched on in February 1946. Like COLOSSUS, it was a huge machine, weighing 30 tonnes, covering nearly 1,400 square metres and using 19,000 vacuum tubes

for its computing functions. When presented with a series of calculations estimated to take 100 engineers a full year to solve, ENIAC produced the answers in just two hours.

Mauchly and Eckert went on to complete UNIVAC for the U.S. Bureau of the Census. UNIVAC 1 was the first computer capable of handling both numeric and alphabetical (alphanumeric) information with ease.

In spite of amazing performances, it was still extremely difficult to switch computers from one task to another. There had to be a way of storing information inside a computer enabling it to cope with whatever type of problem was fed into it.

This new concept became the problem of mathematician John von Neumann, who finally came up with the answer in 1947. This moved computing from an externally programmed machine to a stored-program computer. These techniques were incorporated in a new type of machine called the Electronic Discrete Variable Automatic Computer, or EDVAC for short.

At this stage all these wonderful machines were sitting around humming away to themselves and performing functions at unbelievable speeds. But there were several major problems still to be sorted out. Computers were still in the form of very big machines, which used vast arrays of vacuum tubes, required large storage areas with controlled environments and which were understood only by the research scientists who worked with them. Because the cost of construction was so high, only a few existed and these were owned usually by governments or large organisations.

The next development in this field with far-reaching social effects occurred in 1948. Engineers working for the Bell Telephone Company in the U.S.A. announced they had developed a new electrical component. This was a small three-terminal semiconductor known as a transfer resistor. It was registered under the trademark "Transistor", which was compiled from TRAN(sfer) and (re)SISTOR.

The transistor provided miniaturisation to a degree never thought possible. Suddenly, machines and electrical appliances began to get smaller and smaller. The power and speed of transistor-equipped computers doubled and redoubled. Memory banks became capable of more and more storage even as the physical dimensions of the machines were reduced. Not relying on heat, transistors required less energy to operate. Out went the vacuum tube.

The modern computer was on the way. The days when one computer filled several rooms were over. The first commercially

produced computers were available by 1950. By 1958 more than 1,000 were in use in the U.S.A. and 160 in Europe. By 1961, the year Major Yuri Gagarin became the first spaceman, the world's largest computer – ATLAS – was used to assist both atomic research and weather forecasting.

The size of computers continued to get smaller and in 1965 the minicomputer appeared. Computers were at last down to a manageable size.

Ten more years passed until the first primitive microcomputer kitset became commercially available. Assembling such a kitset required patience, but it was an even greater challenge to get one running. There was a wide variation in standards. Loading identical information into one of these new computers could take a minute on one machine, five minutes on another, and so on. Off-the-shelf pre-packaged programs (or software as it is called) were virtually non-existent.

Despite all these shortcomings the "personal computer" had arrived. In the U.S.A. it currently sells at the rate of more than 20,000 systems a month.

With the development of "silicon-chip" technology came the ability to mass-produce microcomputers. A home system that may have cost $5,000 in 1982 would have had a comparative cost just 12 to 15 years earlier of $150,000 to $200,000. Mass-production has brought costs tumbling down and they will continue to fall until the computer becomes, in its various forms, one of the cheapest and most-used personal appliances.

The growth rate of the computer industry is without precedent. In our entire history no other device originating from the research laboratory has enjoyed such wide use or become so indispensible to many modern-day operations.

We have come a long way from using fingers and toes or a pile of rocks for recording information, so where do we go from here? We will look into the future in Chapter 10.

Chapter 2

Translating computerspeak

First attempts to find out more about computers and computing usually end in total confusion. The biggest obstacle to understanding is not lack of technical knowledge, but the shock of finding out that although computer language sounds like English some of the words you recognise have totally different meanings while many others seem totally meaningless.

You have just met "computerese" or "computerspeak" which you may think cannot be readily understood by mere mortals. At this stage many people dismiss computing as being too complicated to understand, but essentially it is no different from any other trade, science or art with its own vocabulary of technical terms.

To an accountant "reconciliation" has nothing to do with patching up a shattered romance – it is something to do with accounts; a "strainer" is a large post used to support a farmer's fence; teachers recognise "stream" as part of a grading system; sporting types recognise "squash" as a game, not a vegetable; and to an engineer an "hydraulic ram" is certainly not a bionic sheep.

In spite of the fact that words with varying meanings are used around us every day, people who approach computing and meet familiar words – like "memory", "bus", "hardware" and "disk" – which seem to have taken on new meanings, are immediately thrown into confusion. Ask a computer owner to explain them and you are just as likely to end up with a complete mental blackout or, at best, yet another set of words as incomprehensible as the first.

Unfortunately, there is no short cut to understanding computers without knowing what certain words mean, but let's look at them from a non-technical point of view.

THE BASIC TERMS

First we will clear up the difference between a "microprocessor" and a "microcomputer".

A **microcomputer** as far as we are concerned is simply a small computer, such as an Apple, Commodore or TRS-80, to name but three of the more popular makes. Microcomputers can sit on the top of a desk or be carried around from place to place with relative ease. There are even smaller microcomputers which can be carried in your pocket, but this book is more concerned with desktop models.

A **microprocessor** is something inside a "microcomputer". This is the brainbox which controls the processing operations. It is about 1 centimetre square and contains microscopic electronic circuits. We shall look at it in more detail later.

Many people use the terms "microprocessor" and "microcomputer" interchangeably to mean microcomputer, which unfortunately confuses the issue. The same thing has happened with "transistor". Strictly speaking this means an electronic component called a semiconductor, but to most people "transistor" means a portable radio.

Remember, only after microprocessors are combined with a microcomputer system can we get the familiar computer operations of input, output and storage of information. Microprocessors fit inside microcomputers, not the other way around.

Look at a microcomputer and you will see that it is made up from a number of components. These may be combined into one unit or they may be spread around and joined to the main unit by cables.

A typical microcomputer consists of a keyboard, a box of electronic circuitry, a display screen and a disk drive or cassette recorder. These things, plus other similar accessories such as printers, are called **hardware**.

The programs which are put into the machine enabling it to carry out particular tasks are known as **software.**

A **program** is a set of instructions for the computer which gives it the ability to handle accounts, draw pictures, compose music or play games.

An easy way to remember the difference between hardware and software is that if you drop hardware on your foot it hurts!

Buried somewhere in the heart of the electronic circuitry is the microprocessor. This is also called the "Central Processing Unit" or **CPU.** It is the home of the famous "silicon chip" about which we hear so much.

A **silicon chip** is just that – a chip of silicon. What makes it so special is that combined with it are complete electronic circuits so small that they can be seen clearly only under high magnification. The way in which all these circuits fit onto an object so small is an interesting but very long story in itself, so for now just accept they are there.

I once heard a joke which asked, "How do you put two elephants in a matchbox?" The answer was to look at the elephants through a reversed pair of binoculars. This would make the elephants look so small that you could pick them up with a pair of tweezers and pop them into a matchbox. The system used for micro-circuitry isn't very different.

The breakthrough with the "chip" is that hundreds of complete circuits can be combined with it. If you can visualise a truckload of transistor radios being reduced until they fit onto something smaller than the size of a postage stamp, you have the right idea. The bonus is that you still have the same number of complete radios, each doing its own thing.

The introduction of the transistor and the development of miniaturisation has enabled literally thousands of transistors to be combined in circuits within a single silicon chip. This microprocessor controls, interprets and carries out the basic arithmetic, logic and control functions in the microcomputer. It is the CPU (Central Processing Unit).

CPUs, like cattle and dogs, come in different breeds. They vary according to the type of system built into the "chip". The CPU for the Apple and Commodore microcomputers is called the 6502; for the TRS-80 it is known as the Z-80. Other systems include the 6800 and 8080, to name but two of the many types available.

While the machine is operating, information is held in its **memory.** The amount of information that can be held in your computer at one time is limited by the amount of memory pres-

ent. Memory is usually expressed as 16K, 32K, 64K, and so on. The letter "K" is an abbreviation for "Kilobytes", or thousands of "Bytes".

The term **Byte**, which is pronounced "bite" (exactly the same as when you sink your teeth into an apple), is the unit used by computers to measure a piece of information. In general terms, each character of print and each extra space on this page would represent one byte. Now you can see that a machine with 64K of memory is capable of holding up to 64,000 bytes of information at one time.

Information is usually passed into the computer through a keyboard which normally conforms to the standard typewriter layout known as the QWERTY keyboard, so named after the first six alphabetical keys on the top row. Some computers have additional symbols, graphic characters or functions available on some keys, and sometimes a second numerical keyboard similar to a calculator is also available.

Information is usually displayed on a screen. On some machines, such as the Commodore, the screen is built-in as part of the main unit, while others employ a separate visual unit joined to the main machine by a cable.

Some computers use the home television set as their display unit – the output from the computer is fed through the antenna terminals on the television set. The other type of display screen is known as a **VDU**, or visual display unit. This type of screen is operated by the computer and is not capable of receiving television broadcasts. You may also hear a screen referred to as a **monitor**.

Programs and information you may want to use several times must be stored outside the computer. When the computer is turned off all program material in the memory is lost. Information is usually stored on a cassette or a disk.

Cassette storage has the advantage (with most computers) of being able to utilise the same cassette recorder used for music recording and replay. Information is passed from the computer to the recorder by a cable and can be recorded on normal music cassettes. This is the same as recording something off the radio or from another cassette machine. If, after recording the information from your computer, you play the cassette back you will hear a collection of very strange sounds. The computer changes the information into what is called a "digital" form, which is expressed in the cassette as a series of sounds. This permanent record of your program or other information can be put back into the computer by replaying this recorded track, just as you

would replay an ordinary recording. Some computer systems allow you to hear the sound during transfer, but if yours doesn't, the computer will always signal the start and finish of cassette storage or loading.

One disadvantage of cassette storage is that it may take several minutes to transfer large quantities of information.

The most efficient form of information storage is on disk. These are also known as **diskettes, mini-disks** and **floppy disks**.

Unlike the much larger and rigid disks used in major computer installations, microcomputer disks resemble a small version of a 45 r.p.m. record. They are flexible Mylar disks coated with magnetic oxide, just like recording tape, and are enclosed within an outer protective covering. When they are to be operated you place them in a **disk drive**, which is a small rectangular box with a slot at one end covered by a door (see page 31).

Before leaving disks behind, let's take a look at the **hard disk**. These are usually found in large computer installations, although they are also available for use with microcomputers. Unlike "floppies" they are much bigger and can store much more information. A hard disk is made from a finely honed piece of metal which is coated with various magnetic materials. Hard disks are housed in totally protected environments as they are prone to contamination from dust, hair, smoke particles and many other things. Hard-disk systems are very expensive.

The quantity of information that can be stored on disk varies. This is controlled by the **operating system** which decides how the information will be arranged and packed on the disk's surface. Small floppy disk storage can range from 140K to 500K, compared with hard disks whose storage is measured in megabytes, or millions of bytes (see also page 32).

You may have a number of programs or blocks of information, each of which may fill the available memory in your machine. All these different programs can be stored on one disk: you simply call in the one you want to use at a given time. When you have finished with it, it is saved onto the disk and the next program is loaded into the machine.

While information can be passed in and out of the machine and recorded on cassette or disk, you may sometimes need written copy. This could be a letter, invoice or any similar document. To obtain it, information is passed to a printer. The piece of paper taken from the printer with the information typed on it is called either **hard copy** or **print-out**.

Now let's add a little more detail.

There are two kinds of memory present in a microcomputer. These are called **ROM** and **RAM**. ROM stands for "Read Only Memory" and RAM for "Random Access Memory".

It has already been said that when you turn off the computer any information in its memory will be lost. This is correct as far as the information you have put in is concerned, but certain information will always remain. This is the **operating program** which tells the computer what to do when it is switched on, which key has been pressed and how to react to the instructions that have been entered. Such a set of instructions, sometimes referred to as the **system monitor**, contains all the operational procedures and is stored in ROM. This is also where your computer's particular language is stored. ROM is arranged so that it will not be used to store any information you enter into the computer. It is like a reference library into which the machine darts from time to time to look up instructions to see what it should do next, but it leaves no information behind as it returns to the job. Thus we have "Read Only Memory" or ROM.

RAM is the type of memory the machine uses to store information entered through the keyboard, off disk or from some other external source. Information in RAM can be read, re-written or replaced at will, and when you turn the machine off RAM is cleaned out ready for next time. It can be thought of as a general storage cupboard.

RAM is also the type of memory that decides how much material your computer can hold at any time. Many microcomputers are purchased with a nominal amount of RAM – say 16K – which can be increased as the owner can afford it. RAM is usually increased by 16K-blocks. This is why memory is usually expressed as 16K, 32K, 48K, and so on.

All this ROM and RAM is housed under the lid of your machine. If you lift the lid and look underneath you will see rows of flat, rectangular plug-in devices. These are **Integrated Circuits**, usually referred to as **ICs**. If you pull one out you will find that it has a row of metal pins along each side, which gives it the appearance of some sort of futuristic insect. Inside this flat object are some of those miniature circuits discussed earlier. All the RAM, ROM and, indeed, just about everything else is hidden away inside these things.

While you are looking under the lid you will come across **bus** – not something you catch to work in the mornings, but a group of wires or printed circuitry that allows the CPU, memory and other bits to exchange information.

Earlier I explained that "hardware" meant the physical com-

ponents of a computer, while "software" is the program put into a machine to enable it to carry out a particular purpose. There's one more: **firmware**. Firmware is the name given to those programs that live in ROM. Think of them as being "firm", or unchangeable or untouchable.

Data is the raw material fed into the computer for processing and the production of information. This could be a list of names or numbers entered through the keyboard, or electrical impulses from temperature-sensing devices attached to the computer (see page 28). A shopping list is data; so is the time on your watch, a list of names in the telephone directory or the details on a pay slip. Data is any fact, symbol, letter or number that can be used to express a situation, condition or idea.

Data is the raw material used to provide **information** and "information" consists of specific data. Microcomputers will accept **alphanumeric data**, which simply means data consisting of both alphabetical and numerical characters.

Anything that goes into a computer is called **input** and anything that comes out is called **output**. Input can be made by keyboard, disk, cassette or from some other external source, such as the sensing devices already mentioned, or even from another computer. Output usually goes to a screen, printer or storage device, but it can also be transferred to other computers, a paper-tape punch such as that used on a telex machine or teleprinter, or to external devices used to perform some automated function.

SUBDIVIDING SECONDS

It is always being stressed that computers can process large volumes of material in a short space of time, so let's have a look at what this means. In our day-to-day environment most of us adjust our schedule to the passing of hours and minutes. Some groups of people such as athletes and broadcasters are more familiar with minutes and seconds.

Computers start with seconds and proceed to carve them up into further measurements of time. The first of these is the **millisecond**, which is one thousandth of a second, but that's only the beginning. There is also the **microsecond**, which is one millionth of a second. The fastest of all is the incredible **nanosecond**, which is one thousand-millionth of a second – a space of time probably best understood by a computer.

As nanoseconds crop up quite often, especially when computer operation is being discussed by computer speed freaks, let's have a go at understanding what a nanosecond is in realistic terms.

One thousand-million is an American billion, so let's call the

nanosecond a billionth of a second. It is probably still too much for you to comprehend that a computer can perform a billion switching operations inside the space of one second, so let's have yet another try at putting it all in perspective by relating the measurement to a human action. To do this we must step once more into the realms of imagination.

Pretend you are a billionaire who has been out in the sun too long, gone a bit crazy and decided to give away all your money. Despite your obvious mental state you wish to go about this methodically, so you stand on a street corner for eight hours a day, five days a week. As people file past you manage to give away a $1 note every 10 seconds. If you handed out your first $1 note in 1981, how long would it be before you handed out the last one? If you could survive that long, you would be handing out your last $1 note somewhere around the year 3320. Compare handing out a billion dollar notes and turning a switch on and off one billion times and you will have some idea of the amount of work a computer is capable of doing in one second.

MORE "COMPUTERESE" YOU SHOULD KNOW

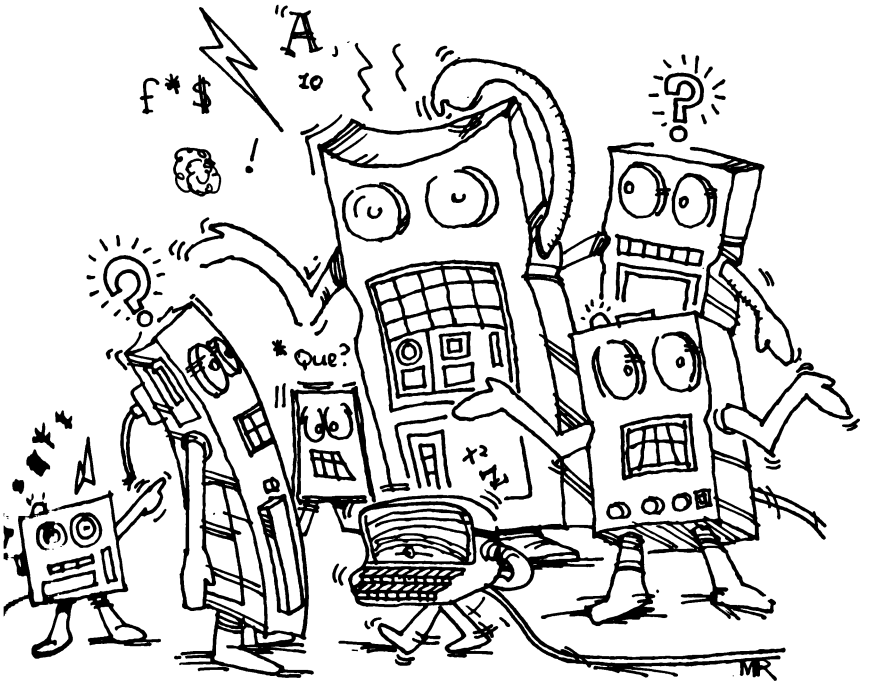
So far we have covered the meanings of a lot of terms connected with computing, but there are still a few words to go.

Although microcomputers do not use **punch cards**, these are an integral part of computer operations and you should be familiar with them. They come in a standard size and contain 80 vertical columns, each column being made up of 12 punching positions or rows. Data is placed on the card by punching holes in particular sequences and positions. Large quantities of cards can then be fed into a machine which reads them and passes the data into the computer.

A similar system used with microcomputers is the **Mark Sense Card**. Instead of punching holes, a special card is marked with a heavy pencil and fed through a **card reader** where it is optically scanned. As with punch cards the data is then passed to the computer. These cards are generally used by educational institutions. (See page 35).

The final group of words to be studied is "Mainframe, Mini and Micro" computers.

Mainframe computers in micro terminology are the large computer installations manufactured by such companies as Burroughs, IBM and Prime. Mainframe computers are definitely not portable and their price tag tends to start at about half a million dollars and works upwards. Access to a mainframe is usu-



ally through a remotely located "terminal", which consists of a screen and keyboard.

Minicomputers are somewhere between mainframes and micros. They vary in size from self-contained but fairly bulky systems to sawn-off versions of a mainframe. Some employ a mainframe for their final storage and for accessing certain data or programs. Their general cost can range from $20,000 upwards.

Microcomputers range from something that fits in your pocket to desktop models. They are self-contained, portable and are operated by a microprocessor.

Remember that terminology changes over periods of time. Certain words take on new meanings until they become common usage. Already the words "minicomputer" and "microprocessor" are being used when what is really meant is "microcomputer".

As you read and learn more about computers and computing, other unfamiliar words will fall into place. Conversations that sounded incomprehensible will become clear to you and the world of computing will cease to be like a mysterious jungle.

Chapter 3

How computers work

Despite any personality that may be attributed to them, computers are simply machines that do exactly as they are told – no more, no less. They are not influenced by anything around them and suffer from total amnesia concerning previous operations, unless that same set of instructions is put back into memory for further processing.

UNDER THE COVER

Many people find it difficult to comprehend how a computer carries out its functions, because – apart from what appears on the screen or comes out through the printer – there is nothing to see. Flashing lights and weird sounds are found only in the movies and are not common features of microcomputers. Oddly, some people who find the fact that there is nothing to see a problem are willing to accept the function of transistor radios or stereo systems without question. Nothing can actually be seen happening with them either and yet sound comes out of the speaker.

Computer users can be divided into four groups:

1. Those who are concerned only with operating a computer and have no interest in its inner workings.
2. Those who are interested in a general non-technical explanation of how a computer works.
3. Those who become involved in writing programs.
4. Those who are mainly interested in the technical aspects of the computer and associated hardware.

To explain the inner workings of a typical microcomputer we shall take a simple non-technical approach which assumes you are neither an advanced programmer nor have a great knowledge of electronics, apart from the ability to change a household fuse.

Rather than a simplification of technology it was the invention of a computer language called BASIC that put computer oper-

ation into the hands of non-professionals. Practically all microcomputers use this language, although some are capable of using other languages as well. However, there is a catch. As in the English language there are variations of BASIC, resulting in the fact that a program for one machine is unlikely to run on another of a different make unless some changes are made to the program. Over the years manufacturers have made additions to standard BASIC according to features available on their own machines, but in order to keep things simple we shall ignore these variations.

BASIC is one of the **high-level languages** invented for humans and is not directly understood by the machine. Deep within the CPU all operations are carried out by using the "binary" system which only has two numbers – 0 and 1. The other numbers – 3, 4, 5, and so on – do not exist. These "ones" and "zeros" represent switches that are either on or off, a concept understood by the computer but sometimes hard for the average user to grasp.

The accompanying description outlines the binary system for those with a curious bent. It is *not* necessary for you to become an expert in the use of binary numbers so, if you prefer to skip this explanation, feel free to do so. This should not disadvantage your general understanding of the process that takes place within the computer.

THE BINARY SYSTEM

Here is a closer look at this binary system and how it actually works. Imagine eight switches on the wall, like this:

SW SW SW SW SW SW SW SW

In binary terms each switch represents a "block" with a particular constant value. These values are read from right to left, starting with 1 and 2, then doubling or, in mathematical terms, "increasing by powers of two". The following table shows each switch and its respective value, and, beneath that, the way the numbers 4 and 30 are represented in the binary system.

SW 128	SW 64	SW 32	SW 16	SW 8	SW 4	SW 2	SW 1	(value)
0	0	0	0	0	1	0	0	(4)
0	0	0	1	1	1	1	0	(30)

Underneath the switches you will see a row of ones and zeros,

which tell the computer whether the switch is on (1) or off (0).

To represent a number, the values of various blocks are added together. In the first row we want the number 4, so only the block with this value is switched on; but the computer must also be told what the other switches are doing, so these are represented in the final figure with zeros to show that they are off. Thus the number 4 is expressed in the binary system as follows: 00000100 (The third switch from the right is the only one switched on.)

Now for something bigger, the number 30. To represent this in the binary system we add the values of various blocks together by turning on the required switches – in this case (see bottom row of table) 16+8+4+2, which equals 30. But we still have to tell the computer what those other switches are doing, so we end up with the binary number 00011110, which also means 30 to the computer. Simply by using the numerals 1 and 0 to indicate whether switches are turned on or off, any number can be represented in the binary system.

Between the insertion of instructions in BASIC, and the binary operations in the CPU, several other steps take place within the computer. These involve the **interpreter** and **assembly** or **machine languages**. (Strictly speaking, the term "machine language" means the binary system we have just looked at, but some people also use the term for what is correctly called "assembly language" (see below). Looking at these steps is rather like looking at a club sandwich with its layers of bread and fillings. Visually, they can be represented like this:

OPERATOR

high-level languages

INTERPRETER

assembly language

machine language

CPU

As the information travels through the layers it is converted at very high speeds into the form best understood at each layer.

First, the operator inserts information in the form of a high-level language or code which he or she can understand (e.g. BASIC). The interpreter changes this into a form understood by the machine – this is known as assembly or machine language.

(The interpreter's role is similar to that of a person providing an instant translation at an international conference. It also does the reverse process, changing the machine output back into BASIC – or whatever language the operator is using – on the screen or printer for the user's benefit. Thus it is kept very busy.)

The number system used at this stage is called **hexadecimal**. It uses combinations of the digits 0–9 and the letters A–F. A programmer who knows how to use assembly language can key in information directly at this stage, without going through a high-level language and interpreter. Unless you intend becoming involved in assembly language programming and hexadecimal number systems, be content simply to know that they exist.

The last stage involves the conversion of hexadecimal into binary numbers, which are used by the CPU to carry out the original instructions.

The whole process can be expressed simply as follows: When a key is pressed the computer gives it a numerical value. This is then interpreted and finally delivered to the CPU in an understandable form. The CPU then carries out the necessary function and the result is delivered by the same procedure in reverse. Any answers or text are either displayed on the screen or passed outside the computer to a printer or similar device.

MEMORY UNVEILED

All input is stored in "memory" – RAM, as discussed on page 16. Memory can be likened to a wall covered with small boxes, similar to the private boxes found at a post office. The number of boxes vary according to the amount of memory available in the computer. If a computer has 48K of available memory there will be, in effect, 48,000 individual boxes waiting to have information placed in them.

Like post office boxes, each memory location (or box) has a number or "address" so it can be found by the CPU, which directs the storage and retrieval of information either from the keyboard or from information specified within a program operation.

PROGRAM WRITING AND HANDLING

Nov we are ready to write a simple program to carry out some calculations. The calculations I have chosen to include are simple random examples. The main point of this exercise is to demonstrate how programs are written to command the computer to carry out such calculations.

In computer programming there are various signs used for mathematical functions. They are the asterisk (*) for multipli-

cation, a diagonal slash (/) for division, the plus sign (+) for addition and the minus sign (−) for subtraction.

We will use the numbers 2, 6, 12 and 15. First this information or "data" will be assigned to "variables".

Variables are names that represent information, in this case numbers. They are also known as variables because the information they represent can "vary" throughout the operation or program.

In this example the variables also represent the places in memory where our calculations are stored and can be found again. Variables also have the advantage of being able to represent long and complex calculations which may be used repeatedly throughout a program. Instead of entering the calculation in full each time, it is assigned a variable at the beginning of a program. When it is due to be used again just the variable is given and the computer checks back to see what it represented. This saves a lot of extra keyboard work and saves using up valuable memory space.

The variables assigned to our numbers will be:

A=2 B=6 C=12 D=15

This program will consist of a **line number** which provides the sequence of events, a **statement**, a **variable** and the **function** to be carried out.

```
10 LET A=2
20 LET B=6
30 LET C=12
40 LET D=15
```

You will notice the lines are numbered 10, 20, 30 40, instead of 1, 2, 3, 4. This is to enable extra lines to be added within the program, such as 15, 22, 23, 24, 32 and so on. If the lines were numbered 1, 2, 3, and so on you would have to rewrite the whole program each time you wanted to make an addition to it.

In lines 10–40 we have "assigned the variables" we will be using. They tell the computer that if it looks into the memory in location A it will find the value 2; B has the value 6; and so on.

Now for the full program, including the functions.

```
10  LET A=2
20  LET B=6
30  LET C=12
```

These are the variables

```
40   LET D=15
50   LET Z=A*B .....Multiply A(2) by B(6): 2×6
60   PRINT Z .....Print out the answer Z
70   LET Z=D/C .....Divide D(15) by C(12): 15÷12
80   PRINT Z .....Print out the new value of Z
90   LET Z= D−A ....Subtract A(2) from D(15)
100  PRINT Z .....Print out the answer Z
110  END .....End of program
```

In line 50 the program says, "Multiply the value called A by the value called B and represent the answer by the variable Z." In line 60 the program says, "Take the value of the variable Z and print it out on the screen."

Each time the computer reads a variable – for example the variable A – it races off to memory and checks what was in the box labelled A. In each case it would find the number 2. You will notice that the value of the variable Z kept changing throughout the program, showing how a variable can be used to represent either a constant or changing value.

Other instructions in the program – such as add, multiply, divide, print and end – are already part of the "firmware" in the machine. When the word **print** is used, it is a word that already stands for a complete routine recorded within the machine. In reality the word "print" triggers off a whole series of machine code which performs the function of printing information on the screen. This bank of machine language tucked away out of sight is often referred to as the "system monitor", as already described on p 23.

The system monitor is the actual program that operates the computer and makes it capable of doing so many things. When you enter a program there are certain key words, like "print", which have a definite meaning to the machine. When it finds one of these words a particular routine in the system monitor is activated. This is similar to starting a car. When you turn the key certain "messages" are sent to the electrical and fuel systems and the engine starts. You don't have to know in detail what these particular messages might be; you just know that by turning the key the engine will spring into life.

Now let's dig just a little deeper into memory and see how it functions.

So far we have been using our own labels (A, B, C and so on) to represent memory locations, but the computer already has its own set of memory addresses, which are not normally considered by the user. In fact, the machine would take the variable

we used and would store it in an area of memory reserved for user operations.

Supposing one line in our program listing read:

```
10  LET J=657
```

When the program was entered into the computer that particular line would have been stored in one of the machine's own memory locations – let's call it location 23.

When the program is "run", the computer works through the program listing according to the program's instructions. When it comes to carry out or "execute" line number 10 it would dive off to location 23 where the line was stored, look up the variable (J) and use the information (657) that it represented.

So far we have used only numbers, but words and even sentences can be assigned to a variable and used in the same way. Words are usually enclosed in quotation marks and are referred to as a "string". For example:

```
"HELLO"
```

"HELLO" is a string, and the quotation marks tell the computer where the string starts and ends.

When using strings most computers require the addition of the dollar sign to the variable to show that a string is being used.

```
10  LET A$="THREE"
20  LET B$="BLIND"
30  LET C$="MICE"
```

A$, B$ and C$ are **string variables**. If somewhere in our program we wished to print the word "mice" we would simply put:

```
50  PRINT C$
```

(50 is a random line number I have chosen for this example.) This line would also contain whatever punctuation your particular machine requires (see page 59).

As with numbers, the string variables could be arranged to print the words in a variety of combinations.

You may have noticed that with this example and the earlier numerical example of variables the same letters have been used. When the computer finds the variable A, the number 2 will be printed. When it finds the variable A$, it will recognise it as a string and print the word "three". The variables A and A$ have entirely different meanings.

Each of the above examples of string variables is written, and would be entered through the keyboard, in BASIC language. The

computer would have to change them into machine language and would store them in various memory addresses or locations in RAM. The program that would carry out these functions would be hidden in ROM and could not be changed by the user.

THE OUTSIDE WORLD

The ability of computers to monitor temperatures, burglar alarms, open and close windows according to weather conditions and so on is quite simply accomplished.

As an example let's take the monitoring of temperatures during a manufacturing process. If the temperature rises above a certain level an alarm is sounded. This whole process is controlled by a program that has been entered into the machine and is sitting in memory. A thermometer is connected or "interfaced" with the computer through an "analogue to digital" circuit board. The thermometer registers temperature in the form of electrical impulses, which are translated through the interface board into a form understood by the computer, and this information is checked by the program. When the temperature rises above a certain level the computer sends an impulse out through the interface board, a switch is closed and the alarm bell rings.

This, in effect, is the same routine that takes place when you walk up to a self-opening door. A beam is broken, a motor is switched on and the door slides aside.

Many microcomputers can generate music which can be played back through an internal or external speaker. This is done by utilising the system-monitor routines which are normally used for "beeping" the speaker when a mistake occurs. A limited range of two or three octaves can usually be generated by varying the pitch and duration of the tone routine resident in the machine. More sophisticated hardware is also available for serious and more elaborate musical composition.

So far we have looked only within the computer. It is now time to have a look at devices which are combined with the computer, such as disk drives, printers and so on.

Chapter 4

The add-ons and extras

In computer terms the various pieces of equipment connected with the computer to allow it to communicate with the outside world are called **peripherals**. This term also covers devices that enable information or data to be passed into the computer and equipment used for program storage.

Generally placed under the heading of peripherals are:

cassettes
disk drives
printers
plotters
card readers
acoustic couplers
modems
audio/voice devices

CASSETTES

Many microcomputer users first employ cassettes for program storage. Most have cassettes which are the same as those used for voice or music recording, although special data-tape cassettes are available.

Some computers have built-in cassette recorders, while others are supplied with a special cable which links the computer with a standard domestic cassette recorder. In some cases a particular make of recorder is required.

In order to store programs on cassette, the program data is passed to the recorder as **digitised information** and is recorded by the machine in exactly the same way as a piece of music from the radio, or output from another recorder is recorded. Loading a program back into the computer is simply a matter of "playing back" the program to the computer, which will then store it in memory ready for operation.

This "saving" and "loading" of programs is usually a manual operation, although some makes of computers operate the cassette recorder by remote control.

In spite of its apparent simplicity, cassette storage has a number of disadvantages.

The first is time. Long programs can take several minutes to transfer and this will become a liability if a program, or part of a program, has to be stored and loaded a number of times in one session.

It is essential that the recording and playback heads on a cassette recorder are kept clean and properly aligned. Dirty heads, while bad enough with a music tape, can turn your program or program data into a collection of garbage as far as the computer is concerned. You may find that your program will not load properly, if at all. Proper head alignment will enable your tape to be read correctly by the recorder.

Some practice with the playback volume may also be needed to ensure a good load. Most computers prefer a specific volume level in order to load properly.

Some computers have a cassette-storage routine which will allow you to enter a name for the program as you store it on tape. When you want to load it back into the machine, just enter the name and the computer will search the tape for it.

If the computer you are using doesn't have this feature you will have to put a voice identification on the tape before you save the program. This will prevent problems later as all computer programs on tape sound much the same.

The biggest advantage of cassette storage is that it is cheap

– a cassette recorder is a fraction of the price of a disk drive.

DISKS AND DRIVES

The most popular and efficient form of storage is on disk. Disks are also known as "diskettes", "mini-disks" and "floppy disks", which have already been mentioned on page 15 although it is worth repeating the following here.

Microcomputer disks look like smaller versions of a 45 r.p.m. music record. They are flexible Mylar disks coated with a magnetic oxide, just like recording tape, and are sealed within a protective cardboard-like outer sleeve. When they are to be used, they are placed in a "disk drive".

Disk drives come in single and double units. On looking at them you will find there is nothing much to see. A single drive is rectangular in shape with a cable from one end connecting it to the computer. Across the front is a slot with a small door and usually a single red light that shines while the drive is operating.

The disk is fed into the slot (like posting a letter) and the door is closed. This action lowers a clamp into the disk's large centre hole, keeping it properly aligned during operation. Controlled from the computer, the drive spins the disk at high speed while a magnetic head travels just above the disk like a gantry crane. This head scans the surface through an opening in the outer sleeve and either **writes** (records) or **reads** (plays back) information as it travels. The head in the drive acts like the record/play head of a cassette recorder.

The surface of a typical microcomputer disk is divided into 35 to 77 separate **tracks**, according to the system being used. Each of these tracks is again divided into **sectors**. Information is stored byte-by-byte within a sector until it is full, and then the operation transfers to the next sector or track.

The operating system also reserves one track on the disk as a directory. As a program or data is stored on the disk, the track and sectors are noted in the directory. When a program is loaded the directory is first inspected to see where the information is located and the head is positioned over the relevant section of the disk.

The biggest advantage of using disk storage is speed of access. Programs that take minutes to run with cassettes can be completed in seconds or less from a disk.

Disadvantages of disks are that they are fragile; they are sensitive to extremes of temperature; their surface is easily damaged by handling, dust or cigarette smoke residue; and, as with music

cassettes, data can be lost or scrambled if the disk is subjected to magnetic fields. Despite this formidable list of drawbacks they have a long life if properly handled.

Floppy disks come in two standard sizes which, despite metrication, are still known as "eight inch" (205 millimetres) and the more common "five inch" (128 millimetres).

On page 15 the hard disk has been mentioned. To reiterate, these are larger and can store much more information than a floppy disk and are more usually found with a mini-computer or a larger mainframe installation. Hard disks are available for microcomputer systems, but they are not usually considered by the average user. Because they are more prone to damage than "floppies", hard disks are housed in a totally protected environment and are rarely handled. They come with the disk and drive in the one unit and are much more expensive than floppy systems.

Storage on microcomputer disks is much greater than the available memory within the machine, but varies according to the type of operating system being used. Small floppy disks range from 140K to 500K of storage, while hard disks may range from 5 megabytes upwards. This means that many programs or data files can be stored on one disk. Each disk also displays an index or catalogue on command, enabling its contents to be verified at a glance.

PRINTERS

If you wish to keep any written records, such as accounts, other business activities or any other type of information, you will need a printer.

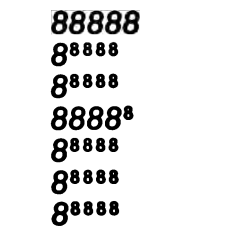
Although there are many different types of printer that can be used with a computer, only three will be discussed here. They are the "thermal", "dot matrix" and "daisy wheel", all within most users' price range.

Thermal printers are almost silent in operation, they are simple to operate and have few moving parts, which adds to their reliability. As the name suggests, they employ an electrical contact to produce an image on heat-sensitive paper. Although offering medium-quality printing, they can also reproduce graphics. Their biggest disadvantage is the special paper they require – it can be expensive.

The most common type of printer available is the **dot-matrix.** The print head is actually a hollow tube containing columns of fine wires which strike through a ribbon to form an impression on the paper. Each wire in the matrix can be fired individually and the various combinations form characters from a series of

dots. The number of dots in the print head matrix can vary, although 5×5, 7×5 and 7×9 are common.

Here is an example of how a 7×5 character is formed:



Selected "dots" in the matrix have struck to form the letter F.

Dot-matrix printers operate at medium-to-high speeds, but they are fairly noisy. The quality of printing ranges from widely spaced and hard-to-read characters to multi-strike print which is clear and easy to follow. Because of the various makes and models it is difficult to be specific about them; however, they all use standard paper which is readily available.

Daisy-wheel printers are the top of the range and produce high-quality printing. Their name comes from the system of printing in which the typeface is set on a wheel. It looks somewhat like a bicycle wheel, complete with spokes, but with the rim removed. At the end of each "spoke" is a piece of type, exactly the same as you would find on a conventional typewriter. During printing, the wheel rotates at high speed, bringing individual letters in line with the ribbon and paper as required.

Daisy-wheel printers tend to be fairly expensive, but they are usually capable of functions such as backspacing, underlining and so on. The actual "daisy-wheels" are available in a wide range of typeface styles and can be changed on the machine very quickly. This makes them ideal for word-processing.

Graphics reproduction can be done only by some makes and models of dot-matrix and daisy-wheel printers and is usually available as an "extra".

When comparing output speeds of different printers the matrix printer is the fastest. Daisy-wheel printers tend to be a lot slower, but give better-quality lettering. The daisy-wheel printer will eventually be replaced by laser printing techniques which will provide top quality with high speed. This will be of particular interest to those engaged in printing and typesetting, but is unlikely to affect the average user in the immediate future.

There are also different types of paper-transport mechanisms available. These are "friction feed", "pin feed" and "tractor feed".

Friction feed is exactly the same as the principle of a conventional typewriter where the paper is held between a roller and platen and is advanced by gear-driven rollers. Problems can arise when several sheets of paper interleaved with carbon paper are used, or when the single sheet of paper is very thin, as it is difficult to maintain good alignment. This can be particularly troublesome if you are using long multi-sheet fan-folded forms.

Paper alignment is no problem with a **pin-feed** sxstem. Metal pins mounted around the roller fit through holes along the margins of the paper, keeping it perfectly aligned as it passes through the printer. Pin-feed systems usually handle only one width of paper.

The final system, **tractor feed,** employs adjustable sprockets so a variety of paper widths can be used. The sprockets slide along a rod and can be clamped into place at any width. The sprockets are driven in such a way that the paper is pulled through the printer a line at a time. This action gave rise to the term "paper tractors".

To help you in selection, here is a guide to the types of printer available and their general uses.

Thermal printers: use where quality is not too important, graphics are often used, the output is fairly low, or funds are limited.

Dot-matrix: for most business applications such as printing forms, data output or correspondence, depending on the quality required.

Daisy-wheel: for word processing and high-quality printing of documents.

PLOTTERS

Although some printers are capable of reproducing graphics, a plotter should be used if high-quality reproduction is essential.

A plotter – which allows the computer to draw graphs, designs and other artwork – has a surface on which a piece of paper is placed. Above this is a pen which can be raised or lowered to the paper surface and can move both horizontally and vertically. Following instructions from the computer, the pen will faithfully reproduce any type of graphic the computer is capable of generating. Some plotters employ different-coloured pens and are capable of changing them during the drawing operation.

Plotters can be used to reproduce graphs, charts, maps, diagrams and other drawings of great complexity. Many line illustrations in books and magazines as well as manufacturing patterns are produced on a plotter.

CARD READERS

These are nothing to do with punch cards. They are more correctly known as **optical mark-sense card readers**.

A card reader is similar in appearance to a disk drive. It is a rectangular box with a slot across the front. In this case, however, when a card is fed into the slot it passes through and is ejected through a similar slot at the back.

The card used in this system is similar in appearance to a punch card, but instead of filling it with holes the user marks small boxes with a heavy pencil or felt-tipped pen. Within the reader the card passes under a number of light emitters which respond to the variations of light passing through the card. When a marked area of the card is read it causes a particular response in exactly the same way as if the information had been entered through a keyboard.

Card readers are mainly used to process educational or research material. In a classroom, cards printed for a multiple-choice answer test can be filled in by students and the results can be quickly obtained by the teacher. Research data of varying complexity can be entered on cards and processed in the same way or it can be forwarded to a central location where the data may be added to a major computer study.

People can easily learn to use the cards with minimal or no training in computer operation.

COMMUNICATIONS

The exchange of information between computers in different locations is now an everyday occurrence. For the personal computer user there are two ways to do this – by "acoustic coupler" or by "modem".

The main components of an **acoustic coupler** are two rubber cups which take the earpiece and microphone of a telephone receiver. The coupler is linked to the computer through a communications card. When transmitting information the computer converts the outgoing material to a digitised series of tones. These are then transmitted through the telephone mouthpiece, over telephone lines and are picked up by the receiving computer with similar equipment.

This is the simplest form of communications device as no special wiring to the telephone is needed and the acoustic coupler is portable.

Unlike an acoustic coupler, a **MODEM** is a permanent device which is wired to the telephone system. It is a more reliable method of transmission than the acoustic coupler and is not affec-

ted by the acoustic interference found on some telephone circuits. The word "MODEM" is an acronym for MOdulator-DEModulator.

In order to use either of these devices your computer will also require a communications interface card and a special program to control the whole operation. Some of these communications programs have the capability of storing the data coming from the MODEM into memory and then transferring it to disk or printer, of automatic phone dialling, of answering, of storing the data and then terminating the call, to name only a few of the features currently available.

The communications capability of a microcomputer can also be used to get access to information from data banks, to transfer information from one office to another or to become involved with the many user networks around the world.

ACOUSTIC OUTPUT

There are peripherals available which allow the computer to provide an output of music or speech.

Musical programming and playback is usually done through a plug-in "card" which interfaces with the computer and contains its own built-in software. This enables anyone to write or compose music and have it played back through an external speaker or stereo system. Most of the available music cards feature "multi-voice", which allows complex musical arrangements to be composed and performed.

Voice recognition and reproduction is still in the development stage. There are software programs that allow voice input to be digitised and stored. This enables certain words to be used in other programs, such as a "talking calculator" for mathematics exercises. Other hardware devices are available which "recognise" words and then reproduce them as simulated human speech.

Other peripherals include the following: "real time" clocks, enabling the computer to keep track of calendar time and allowing the user to have the computer perform fixed-time functions; digitisers that scan maps, pictures and other visual images and convert them to data; camera and video digitisers which enable video information to be entered as data; analog-to-digital converters for monitoring atmospheric conditions, manufacturing processes and medical information; and "game paddles" or "joysticks" used in playing computer games similar to the games systems available for home television sets.

The wide variety of peripherals available gives the microcomputer the power to undertake an endless range of tasks.

Chapter 5

What to do with it now you've got it

The various things that can be done with microcomputers are called "applications" and these can be arbitrarily divided into "serious" and "recreational".

Under the "serious" heading are business, professional, educational and personal applications, while "recreational" takes care of the lighter moments when the computer is used for playing games.

Some computers are purchased with just one application in mind – the buyer is perhaps not aware of the wide range of possible uses. Sometimes computers are bought on impulse, perhaps after the buyer has seen one game, application or program that was very impressive. In the latter situation, particularly, there could come a time when the initial novelty wears off and the computer, like a discarded piece of expensive camera equipment, is left gathering dust.

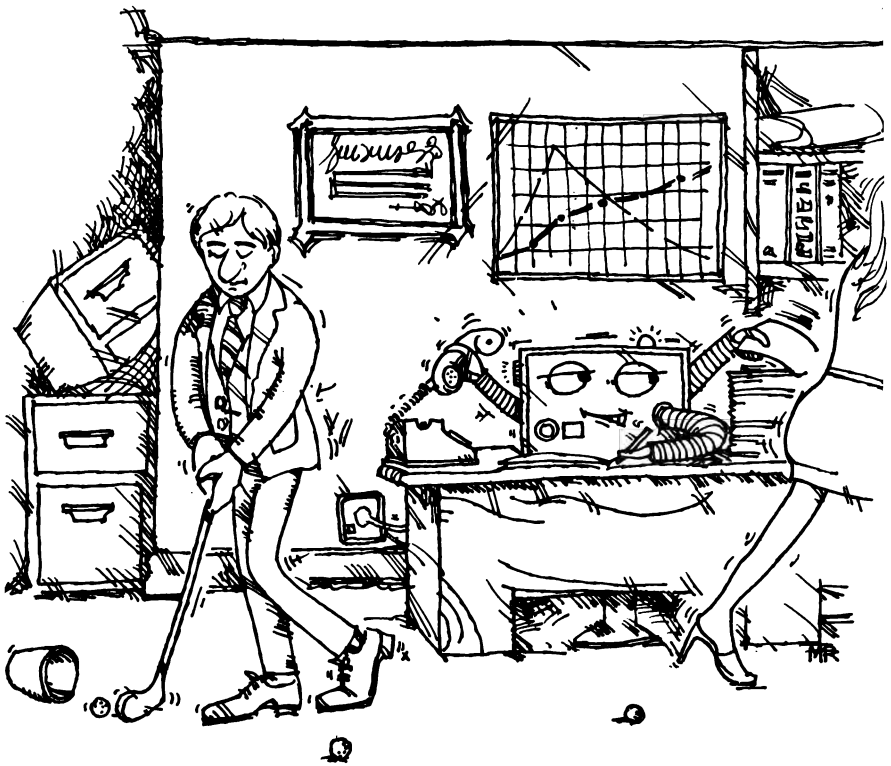
Even if a computer is intended for one particular purpose on a full-time basis it is certainly worthwhile knowing what else it is capable of handling. This chapter looks at many possible uses for a microcomputer.

For convenience, the various applications are grouped under different headings, although a single application may rightly belong under several. For example, tax calculation is not of interest only to businessmen.

Remember also that in order to carry out any of the following applications the appropriate software is needed. Without the right program your computer will not know what to do. It does not handle tasks automatically.

MINDING YOUR BUSINESS

The business world has found that microcomputers, once viewed only as "hobby machines", have opened up the way to increased efficiency and production. Even those with access to large data-



processing systems have found that the installation of a microcomputer enables them to handle much of the preliminary work in a more systematic manner.

Many banks are now using microcomputers for processing things such as loan applications and for printing out the forms as required, which has also solved printing and stationery problems. The accumulated data is later fed into their mainframe systems for inclusion with other daily financial processing.

Companies not able to afford the large price tags of minicomputers or large systems have found they can install microcomputer systems and expand them as financial resources allow. Even business organisations with their own mainframe systems have installed microcomputers as "work stations" as a method of avoiding some of the problems that arise with timesharing operations.

Here are some of the uses for microcomputers in the business world:

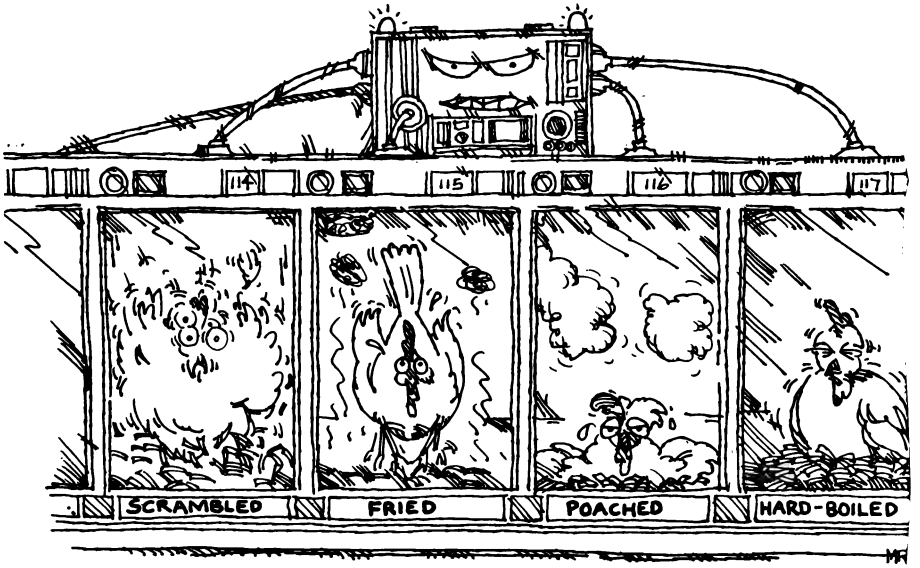
Stock market analysis
Bond yields
Stock options
Mortgage analysis
Financial planning
Loan calculations and options
Options analysis
Project costing
Break-even points
Accounting
Tax calculation
Mailing lists
Word processing
Payroll
Electronic mail
Appointment listings
Bonus calculations
Company data base
Sales records
Sales analysis
Sales forecasting
Inventory control
Project management
Servicing records
Commodity listings
Commodity analysis
Maintenance records
Training programs
Small business budgeting
Plant depreciation schedules
Loan amortisation schedules
Return on investment
Capital purchase budgeting
Investment analysis
Production forecasting
Operating profit margins
Ratio analysis
Corporate planning
Printing mailing labels
Form letter production
Debt collection
Travel planning
Staff records
Professional office management
Generating price lists
Market research analysis
Sales commission records
Point-of-sale recording
Scheduling
Inter-city communication
Accessing data bases
Business simulations
Rental property management
Insurance analysis

Many of these applications do not apply only to city-based offices – together with other uses they are also suited to other occupational groups.

ELECTRONIC FARMING

The agricultural industry is extremely complex as it combines business, practical farming skills and scientific applications on a day-to-day basis.

Some aspects of farming depend on the compilation of data over a long period – for instance, livestock or breeding records – and it is in this role that microcomputers are already part of some farming operations. Other farmers use them for filing technical information or in planning.



A major use of the microcomputer is in the agricultural advisory area. Consultants, who spend much of their time doing routine or time-consuming calculations, or hunting for data that has been filed away, have found the micromputer enables them to streamline many of their routine tasks.

Some of the many agricultural applications are:

Suppliers' accounts
Balance sheets
Cheque book analysis
General cost analysis
Gross margins
Estate planning
Mortgage and loan options
Livestock records
Pedigree records
Production and weight records
Calving/lambing/farrowing records
Health and veterinary records
Performance recording
Soil status
Paddock records/histories
Crop yields
Seed rates
Crop monitoring
Creditors/Debtors Accounts
Pasture growth curves
Market price analysis
Farm machinery depreciation
Machinery maintenance records
Poultry environmental planning
Meteorological records
Project analysis
Bank statement analysis
Labour costing
Forecast budgets
Monthly cash flows
Livestock reconciliation
Breeding histories
Production forecasts
Condition scoring records
Fertiliser requirements
Irrigation scheduling
Crop variety performance

Pasture pest records
Glasshouse environmental control
Feed budgeting
Dry matter production calculations
Least-cost feed mixes
Nutrient requirements
Spray information
Crop costing and planning
Feed use and conversion ratios
Ration formulation
Feed conservation
Rotational grazing planning
Production costing
Long-term planning simulations

CHIPS AND CHALKDUST

The microcomputer has already proved itself as a powerful educational tool. Because of its versatility it can fill a dual role, both in school administration and in the learning process.

Children who already regard television and space flight as routine have no problems adapting to microcomputers and have greeted them as a more exciting style of learning than the traditional process. With a microcomputer many lessons can be presented as "games", which attract even those pupils who find it difficult to keep their attention on subjects being discussed in the classroom.

A pupil whose mind would go blank when asked to calculate trajectory paths is willing to do the task when it is presented as a game. In this case microcomputer graphics are used to show a stunt rider speeding up a ramp to clear a row of barrels. As more barrels are added the angle of the ramp must be changed so the rider can clear them. The reluctant pupil will happily participate in this exercise, which in reality teaches the calculation of trajectory paths in an entertaining form.

In the educational field, microcomputers can be used for:

Programmed learning
Computer assisted instruction
Computer managed instruction
Pupil problem solving
Student information
Timetabling
Stock records
Diagnostic and remedial work
Guidance and counselling
Reading
Comprehension
Drafting
Video animation
Geometry
Computer programming
Computer simulations
Mathematical modelling
Practice and drill
Attendance records
Class information
School accounts
Student assessment
Special education
Industrial drawing
Design
Video titling
Computer art
Computer studies
Computer science
Foreign language studies

LABORATORY AND PRODUCTION-LINE APPLICATIONS

Historically, the computer is identified as belonging to the scientific and technical community, but the advent of the microcomputer brought it into both the laboratory and workshop.

Many tasks have been streamlined with the provision of a microcomputer with "hands-on" capability (meaning having one available for use yourself when and as often as required). Problems are now resolved as they occur instead of being delayed until computer time becomes available.

Microcomputers have also been linked to monitor laboratory and manufacturing processes, eliminating the boredom of continually watching a repetitive function.

Some of the areas where microcomputers have proved useful are:

Computer-aided design – Architectural, engineering, electrical, manufacturing
Energy efficiency
Energy monitoring
Air-conditioning requirements
Home-heating requirements
Laboratory automation
Data analysis
Statistical analysis
Questionnaire analysis
Structural analysis
Sampling
Graphics
Modelling
Logic circuit analysis
Simulations
Machinery control
Environmental monitoring
Scientific calculations
System-reliability studies
Data bases
Probability testing
Quantity estimates
Systems analysis

THE COMPUTER AT HOME

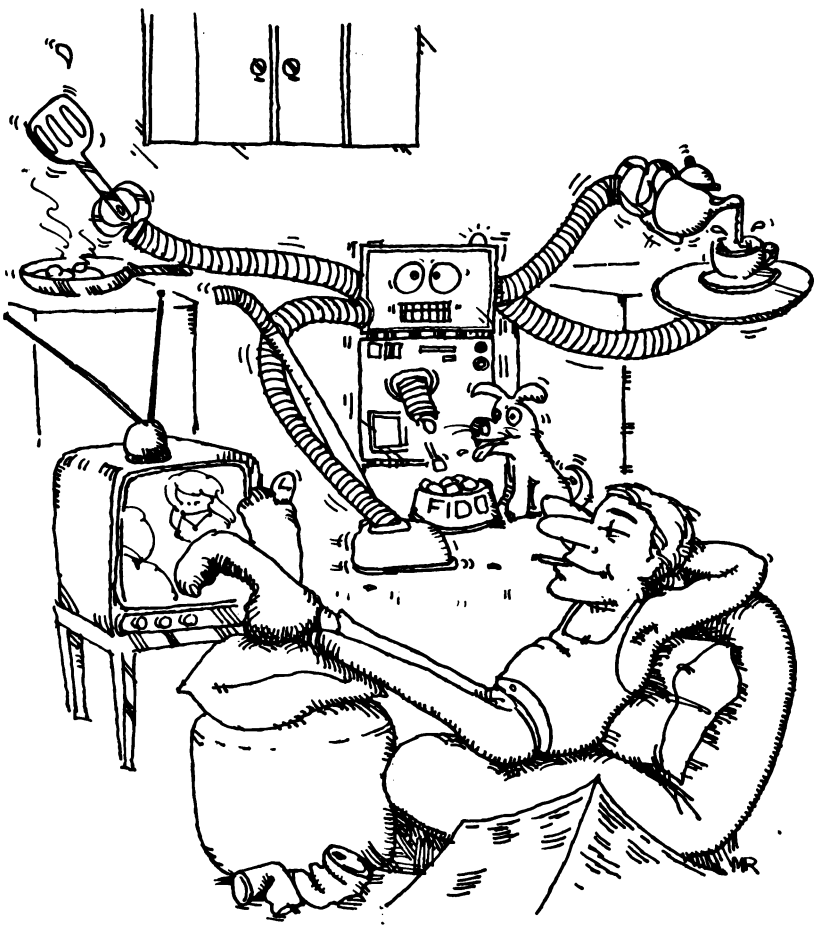
Microcomputers are often called "personal computers" because the early models were originally intended for the home and hobby market. Accelerating technology soon extended their capabilities until they became more powerful and versatile than many mainframe computer systems. In spite of this, it is the personal or recreational role that caused the microcomputer industry to grow at a terrific rate in just a few years.

Computers could now be used by "ordinary" people and it was not long before clubs and user groups were formed. Computer programs became available on just about every subject under the sun, and every new issue of a computer magazine suggested yet more uses for these machines.

Earlier, computer applications were divided into serious and

recreational. It is time now to take a closer look at what constitutes recreational applications.

Some people believe that recreational suggests frivolous uses and they suggest a further breakdown of applications into "personal enrichment", "community applications" and "amusements". Depending on how seriously you take life, these word games could be played forever, so let's just call them "personal applications" and "games" to keep it simple.



Once again the list could run for many pages, but some of the more common personal applications are:

Long-term budgeting
Personal financial management
Property inventory for insurance
Calculating monthly payments
Family health records
Vehicle running expenses
Birthdays and anniversaries
Garden care and planting information
Meal and diet planning
Family genealogy
Astrology
Useful information file
Supermarket lists
Computer art
Astronomy
Amateur radio
Collection Index
Letter writing
Club mailing lists
Ecology simulations
Athletic performance records
Compiling election returns
Graphics animation
Personal improvement programs
Investment records
Address and phone lists
Biorhythms
Recipe index and calculator
Home handyman calculations
Music synthesis
Model railway control
Library or record index
Fishing information
Club newsletters
Computer cryptography
Sports team draws
Sports handicapping
Sports scores
Sports team ranking

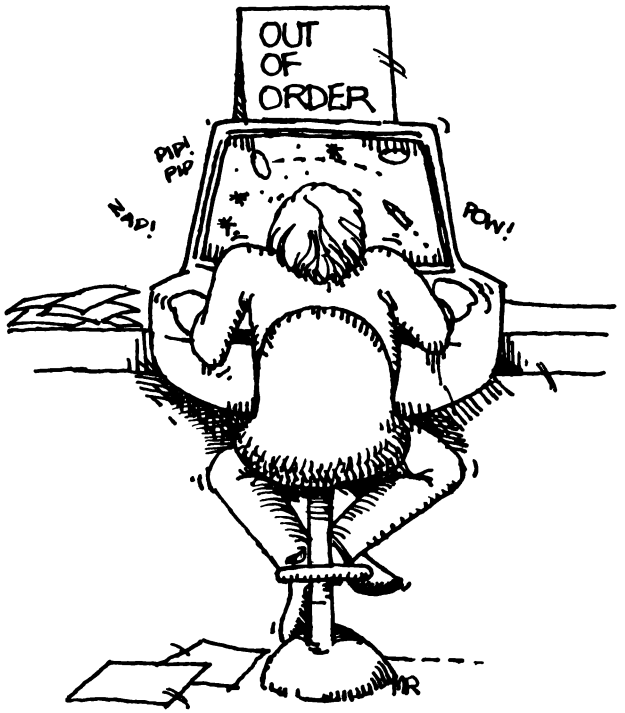
JUST HAVING FUN

Mention a microcomputer and many immediately associate it with games. There are literally thousands of games available, from simple "arcade games" to highly complex and challenging "strategy" or "role-playing" games which may take hours, days or even weeks to complete.

Many games offer different "levels" of play, with increasing difficulty at each stage. Some chess games, for example, start at novice level and work up to international standards of play.

Computer games can be divided into several groups. The first of these are known as "arcade games", the type found in fun-parlour slot machines. They feature plenty of action and whatever sounds that can be provided. These are the games where you shoot down space aliens, go for a high score in a shooting gallery, or take the wheel of a racing car. Many of these games also have a built-in time limit that adds an edge to the action.

"Board games" are computer versions of many old favourites and a few new ones as well. Chess programs have been around for quite a while and improve with every new release. Backgammon, draughts (checkers), cribbage and many more are now available on computer. Many of these programs set a high stan-



dard because of yearly national and international contests being held where not only programs and people but programs and other programs are matched in serious play.

"Sports games" are already familiar in their simplest form through the small units that can be attached to home television sets. Basketball, tennis, hockey, badminton and squash are a few of the games available. In more advanced computer versions, graphics, sound effects, scorekeeping and handicapping may be included for various sports.

The introduction of the microcomputer also opened up the way for "role-playing" games. Many of these could be found on large computer installations and, once memory capacity of smaller machines was increased, they soon became generally available.

In these very popular games the players actually take part. The most popular of all is the "adventure", a risky undertaking full of challenges and surprises. Usually the player moves from location to location with the objective of amassing points or attaining some other goal. The game has many built-in penalties and a wrong decision could be fatal. These games are actually puzzles because many clues may have to be put together to work out the next move.

Another type of role-playing game is where the player becomes the commander of a space ship engaged in a battle or some other hazardous mission. Each decision that the player makes will affect the outcome of the next move.

"Strategy games" are very popular because many people think they offer the ultimate challenge. Playing any of these is similar to "wargaming", another type of recreation which enjoys a large following.

A typical strategy game may involve a battle where the player commanding a side will have to decide things such as deployment of personnel, rations, ammunition, type of weapons and a host of other detail. Once the battle has been fought the player may have the option of commanding the opposing side and playing again under a totally different set of circumstances.

Some strategy games re-enact famous land and sea battles. "Big-business" games which run the full gamut of corporate politics and intrigue are also available.

★ ★ ★

Microcomputers can be put to many uses, serious as well as recreational – all on the same machine.

You may hear that the number of uses for a microcomputer is limited only by the imagination. That's a fine sweeping statement, but if you are thinking of buying your own machine it can be very misleading. A particular make or model of microcomputer may be limited in its applications because of memory size, lack of graphics, supporting peripherals and available software. But once you are familiar with a particular machine's strengths and weaknesses and know to what extent imagination can play a part, you will still be suprised with the number of things you can find to do on your chosen model.

Chapter 6

Looking at languages

If you want a computer to carry out a particular task it must have a program that tells it how. That program must be written in a language the computer understands, and the programmer must be able to construct a set of instructions that follow the rules of the language being used.

Why are special computer languages needed at all? Why not just write them in English and make everything so much simpler? The answer is that computers are very exacting machines and cannot think for themselves, therefore they must receive very precise instructions.

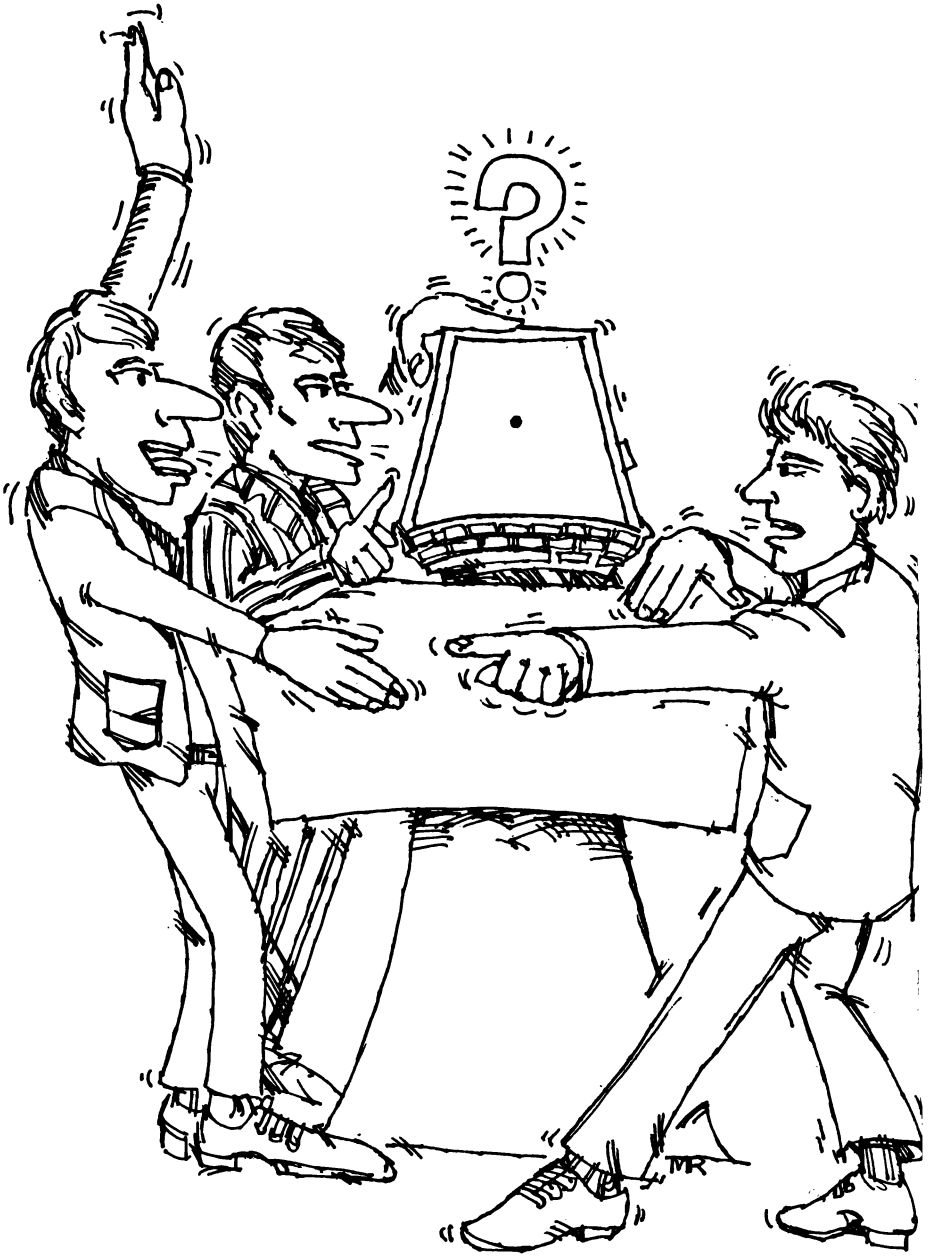
Take the word "fit". Does it mean to be the proper size or shape; to adjust or alter; to insert into place; to be healthy; to have a convulsion; or does it mean to equip?

English, being a very imprecise language, was never used at all in the early days of computer operation. It is only in recent years that a number of languages, known as "high-level languages", have evolved. These are written in English, but to a set of strict rules which ensure that ambiguities are not present in the instructions given to the computer.

Computer programming in its original form was done to conform with the method used by the computer to store its programs. This was in the form of "binary numbers" (see page 22).

Programs written in this "machine language" had to be carefully constructed as a series of ones and zeros, an extremely tedious and time-consuming process which was subject to many errors. Today's machine-language programs are usually written with "Octal" or "Hexadecimal" numbers, which are far more convenient to use. Using the hexadecimal system, a piece of programming would look like this:

```
0BC8 – 09 0B A9 8D 4C D0 03 8D
0BD0 – 8D A0 A0 D2 C5 D7 D2 C9
```



Computers cannot think for themselves. The program must be written in a language suited to the particular computer, and the programmer's instructions must follow the rules of that language.

This method was also fairly time-consuming, so an alternative programming method was devised which consisted of representing the machine-language instructions in a symbolic form. It is known as **assembly language programming** (see also page 23). With this technique the operating codes used by the computer are expressed as a series of letters. To help the programmer remember them and to indicate the meaning of each instruction, they are expressed in a mnemonic form such as the words "ADD" or "JMP" (jump).

A piece of assembly language programming looks like this (it is not necessary for you to understand the meanings of these symbols at this stage):

```
0302    16  CATLG    EQU   $A56E
0302    17  HOME     EQU   $FC58
0302    18  MONITR   EDU   $FF69

0302    25           LDA   #$4C
0304    26           STA   VECTOR
0307    27           LDA   #START
```

When a program is written in this form it is translated into machine language by a program called an "assembler" which also checks it for obvious errors.

Although this is much faster than programming directly into machine language, assembly programming still requires the programmer to be familiar with all the operating codes and the address or storage locations in the computer's memory.

Despite the availability of high-level languages, many programs are still written in assembly language. To confuse the issue, some people use the terms "machine" and "assembly" interchangeably, so let us refer to them as **low-level languages** to avoid any confusion.

Low-level languages, such as the example reproduced above, do offer some advantages. They tend to take up less memory than many of the high-level languages – an important feature in a computer with limited memory – and an experienced low-level language programmer can write programs that will run much faster and more efficiently than the same programs written in a high-level language.

High-level languages reduced the amount of time needed to write programs and helped reduce the possibility of errors by the programmer as they were much easier to write and check.

Although it is much more flexible than programming in machine or assembly languages, programming in high level also fol-

lows a set of rules. Special sets of instructions, words and numbers must be used as well as very exacting forms of punctuation.

Like any programs written in assembly language, a high-level language program must be translated into machine language before it can be understood by the computer. This procedure is carried out by the computer, using an "interpreter", and is not apparent to the computer user (see page 23).

With the number of high-level languages available the thought may occur that some must be better than others. An attempt to follow this line of thinking will end in total confusion, but will stimulate lively debates among computer users. The subject of "best" languages is better left alone. It is similar to asking, "What is the true religion?" or, "Which is the best football team?".

High-level languages such as COBOL and FORTRAN have been used for some time, but the language used by nearly all microcomputers is known as BASIC.

BASIC (Beginners All-purpose Symbolic Instruction Code) was developed in 1973 at Dartmouth College in the U.S.A. by Professors John Kemeny and Thomas Kurtz. Their original intention was to devise a simple language for non-professionals who wanted to write their own computer programs. Many of the Dartmouth students used the college's time-sharing computer and the language was originally intended for them.

BASIC is a simple language which enables the user to write, examine, modify and correct programs with a minimum of fuss. In a short time its popularity grew, and BASIC soon had a large following outside Dartmouth College. Here, at last, was a computer language designed for ordinary computer users. Its adoption as the main language for the personal microcomputer was a terrific breakthrough, and the use of BASIC became worldwide.

The popularity of the language has instigated the evolution of many "dialects". Manufacturers of microcomputers have either modified the original version or added their own particular sets of instructions to the language. Because of this no two versions of BASIC are exactly alike, and a program written for one make of microcomputer is unlikely to run on another, even though they both apparently use the same language. Although the differences in other BASIC programs can be understood by the non-professional programmer, extensive changes may have to be made before a program can be run on another machine.

Despite all these apparent shortcomings, BASIC is easy to learn and simple programs can be written after only a few hours' study. Most microcomputer operating manuals give examples of

BASIC programming and these can be used as an introduction to the language.

There are many other high-level languages, and some of them can be used in microcomputers, either as a plug-in piece of hardware or in a disk form which is compatible to the particular make of computer.

COBOL (COmmon Business-Oriented Language) was developed in 1959 and was devised to process alphabetic as well as numeric data. It has since become the most widespread high-level computer language in the world (except for BASIC) and is used in many large installations.

COBOL was intended for use where large volumes of business information could be entered in a language that resembled English and then be converted to machine language. One of the great advantages of COBOL is that programs written for one computer can be run on another with minimal, if any, modification.

Another high-level language, this one developed for scientists and engineers, is **FORTRAN** (FORmula TRANslation). Although first used in the mid-1950s, it was not until 1966 that it was finally standardised by the American National Standards Institute.

FORTRAN has a heavy mathematical approach. Although today it is sometimes considered to be unwieldy and somewhat inefficient, it remains popular because of the large resources of engineering and scientific programs still available. FORTRAN is also used for some commercial and educational applications.

Intended to be an international language for computer manufacturers, **ALGOL** (ALGOrithmic Language) was introduced in 1958. It was also intended for use with scientific and mathematical projects.

ALGOL was developed through a combined German and American group, but it never achieved much popularity with computer manufacturers. This was mainly due to the fact that in its original form many symbols, not normally found on input and output devices, needed to be used. Later improvements overcame this problem, but ALGOL's popularity remains mainly confined to Europe.

One of the high level languages that is gaining increasing popularity among microcomputer users is **Pascal**. It was developed in the late 1960s by Dr Niklaus Wirth of the Federal Institute of Technology, Zurich, Switzerland, and is named after Blaise Pascal. Wirth's intention was to correct the faults in some of the other high-level languages such as FORTRAN, COBOL and BASIC.

Large programs can be written more easily by first constructing a series of smaller ones. This is called "structured programming" and consists of selecting several key concepts, developing the program systematically, having good documentation and ensuring good program flow. Pascal was designed around this concept, and programs written in Pascal can perhaps best be described as consisting of a number of mini-programs. It is thought that Pascal and BASIC may well become the two major languages for personal microcomputers.

Another high-level language proving popular for classroom use is **PILOT** (Programmed Inquiry Learning Or Teaching).

Developed at the University of California in San Francisco, PILOT is an extremely simple language intended for computer-aided instruction (CAI). As a computer language it does not handle complex problems very well, but it is designed rather for classroom drills, tests and conversational keyboard operation by pupils.

One of the high-level languages that has never achieved much popularity because of its complexity is **PL/1**. This language was introduced in 1965, but it has never been widely accepted. As many as 200 keywords may be required for a typical application, but despite this handicap it has proved reasonably popular for handling scientific applications.

A programming language with a fast-growing and enthusiastic user community is **FORTH**. This uses concepts very different from other high-level languages and was developed for control applications, data bases and business use. Invented by Charles Moore, FORTH is a very flexible language and gives users complete access to their machines. Programs use little memory and run much faster than some of the other high-level languages.

Other high-level languages include LISP (LISt Processing language); APL (A Programming Language); and SNOBOL (StriNg Oriented symBOLic language).

You may never consider using anything but the BASIC language that comes with your computer. You may never see a program listing in any of these other languages, but some of them can be used on microcomputers and it is handy to know that they do exist.

Chapter 7

About programming

A computer program is like a recipe for cooking a cake, but it is more detailed. A recipe for a cake tells you what ingredients are required, the amounts and how to mix them together. Finally, it gives cooking instructions.

A computer program aimed at solving a problem first lays out the "ingredients" of the problem, then tells the computer the "method" to be used to solve it and finally states what to do with the answer when it is obtained.

In a cake recipe there are many things left unsaid which *must* be provided when you are writing a program. For example, the cake recipe does not include instructions on how to get a mixing bowl from a cupboard or whether the mixture should be stirred in a clockwise or anticlockwise direction.

Consider how a computer would view getting into the car and setting off for work in the morning.

1. Open garage door.
2. Open car door and get in.
3. Place key in ignition and turn until engine starts.
4. When engine starts, let go key.
5. Engage reverse gear.
6. Back out of garage.

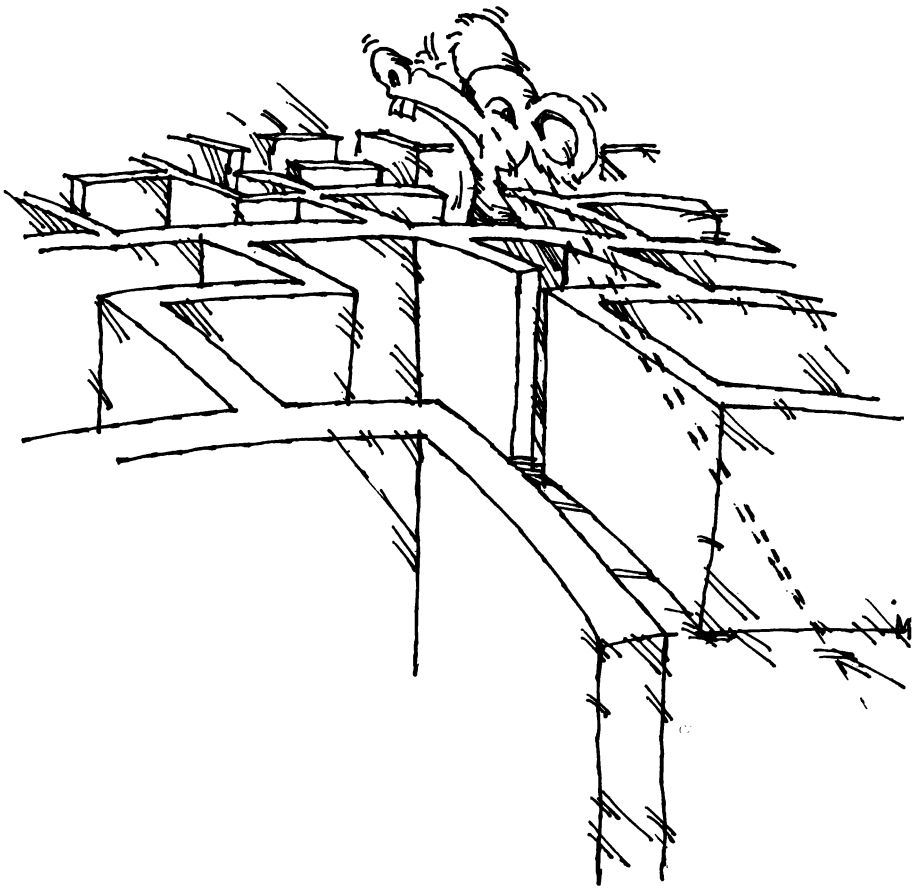
What a pity that the car door was torn off when it was backed out. The instruction to close the door was missed. There was no check to see whether the car was automatic or manual: if it was manual the gearbox would have been ground up as there was no instruction about using the clutch.

There is another often-used example along the same lines which deals with the simple act of cleaning your teeth. Ask people to describe the procedure and you will find that most will forget to mention taking the top off the toothpaste before they squeeze it onto the brush.

These examples illustrate how carefully programming must be approached. If you miss any part of a procedure, however small, the computer will go merrily on its way, creating all sorts of problems and erroneous information.

Remember, "There is no such thing as computer error!"

"Computer errors are created by people."



First step is to define the objective....

Except for some hardware failure, 99.9 per cent of computer errors are caused by people through faulty programming or such things as mistakes when keying in information. Programming is an acquired skill and the more you do, the better programmer you will become.

You do not have to be some sort of genius to write a computer program. The secret is to take it carefully, step-by-step according to your ability and be careful not to get out of your depth.

The first step in programming is to think about what the program is supposed to do and what results are required. Having settled that, you will then be ready to design your program. Properly designed programs are much easier to code and are less likely to be attacked by the dreaded program "bug" that plagues those not well thought out. Program bugs cause things to be left out, errors in logic, faulty results and just about every other mistake you can think of. They have been known to reduce programmers to quivering wrecks. Bugs are hard to find, but they can be traced back from their effects on your program.

There is no mystery to program design. It simply means laying out your program step-by-step until you are satisfied with it and then putting it into coded form for the computer to understand.

PROBLEM SOLVING – STEP-BY-STEP

As with any form of creative activity, there are many ways programs can be designed and no two programmers are likely to operate in exactly the same way, even if the results are identical. There is no "correct" method, but some techniques are better than others.

The design of a computer program can be broken down into several steps:

1. What is the intention of the program?
2. Note the stages needed to achieve this intention.
3. Write them out step-by-step.
4. Break each step down until you cannot go any further.
5. Lay these steps out graphically.
6. Group them into modules.
7. Break them into a control section and subroutines.
8. Go through the logic and make sure it works.
9. Convert the program into code.

Now let us apply these steps to a particular problem.

Joe, a young man who always tries to keep up with the latest trends, has been offered a water bed at a bargain price. Although

it will just fit into his small bedroom, Joe's apartment is located on the second storey of a rather old house which shakes during strong gusts of wind. He knows a water bed is very heavy and wonders whether the floor will support it safely. To complicate matters, all measurements have been given to him in feet and inches. Joe understands only metric measurements, so as well as solving the problem of weight he must also deal with conversions of measurements. While this could easily be worked out on a pocket calculator, we shall approach it as a computer programming problem.

Stage 1

Problem

How much water does Joe's water bed hold and what will it weigh when filled?

Procedure

1. Work out the volume of the bed.
2. Work out the volume of water it will hold.
3. Work out the weight of the water.
4. Compare the weight of the water with the maximum permissible weight loading on the apartment floor.

Known data

a. The length of the bed is 6 feet 6 inches.
b. The width of the bed is 4 feet 6 inches.
c. The height of the bed is 3 feet.

Formulae

(i) The volume of the water bed is worked out by multiplying the length by the width and the height:

$$\text{Volume} = L \times W \times H$$

The answer will be expressed in cubic feet.

(ii) 1 cubic foot of water is equal to 6.25 gallons.

(iii) 1 cubic foot of water weighs 62.5 imperial pounds. This information can be obtained from reference books.

(iv) To convert gallons to litres, multiply by 4.546. (1 gallon = 4.546 litres.)

(v) To convert pounds to kilograms, multiply by 0.4536. (1 pound = 0.4536 kilos.)

Now that this information is noted, the entire outline of the program is written down step by step. Each element of the problem is assigned a "variable" which is used to represent it in the program.

Stage 2

(100) Water bed problem

– The program must first calculate the volume of the bed. Any measurements must be expressed in terms of decimals as computers do not handle fractions (e.g. 6 feet 6 inches is expressed as 6.5 feet).

(120) The length of the bed (6.5 feet) is assigned the variable L.

(130) The width of the bed (4.5 feet) is assigned the variable W.

(140) The height of the bed (3 feet) is assigned the variable H.

(200) The volume of the bed is assigned the variable C. (The volume will be expressed in cubic feet.)

(210) The formula for working out the volume is $C = L \times W \times H$.

– Next the volume must be expressed in terms of liquid capacity.

(220) The number of gallons of water the bed will hold is assigned the variable G.

(230) The formula for working out the capacity is $G = C \times 6.25$.

– Then the weight of this capacity of water must be calculated.

(240) The imperial weight of the water is assigned the variable P.

(250) The formula for working out the weight is $P = C \times 62.5$.

– The capacity in gallons is next converted to litres.

(260) Litres are assigned the variable LT.

(270) The formula for converting gallons to litres is $LT = G \times 4.546$.

– The weight in pounds is next converted to kilograms.

(280) Kilograms are assigned the variable KG.

(290) The formula for converting pounds to kilograms is $KG = P \times 0.4536$.

(300–320) The results of the problem are displayed and printed out in the desired manner.

(330) End of program.

Now the weight of the filled water bed can be compared with the known permissible weight loading of the apartment floor.

This all looks rather long-winded, but it spells out exactly what is required, step by step. Once you are satisfied with this list of instructions, the program can be coded. (The numbers in brackets refer to line numbers in the program listing. They have been inserted in the above list of steps so that later you can relate each line of the program back to the steps in the planning stage.)

CONVERTING THE STEPS INTO A PROGRAM LISTING

In the program listing that follows there are several things that need explaining. The word REM means "remark" and is for the

benefit of the programmer. Anything on a line beginning with REM is ignored by the computer. The sign (*) means to multiply. Instead of 3×2 we use 3*2.

The insertion of plenty of REMS throughout the program can be important during the development stage in order to avoid wasting time later trying to remember what a particular block of code is supposed to do. These "remarks" can always be taken out later if you wish. Line numbers usually increase in increments of ten, which allows for later additions to the program without having to renumber the existing lines.

In the following listing we shall start with line number 100; the calculating section will begin at line 200; and 300 onwards is reserved for the output. As explained above, this gives us room to add to the various sections later if required.

Now we shall progress from the planning stage of the program into the actual coding stage. This is where you "translate" your instructions into the code used by your particular machine so they can be understood and acted upon. This listing is in one of the versions of BASIC language. As you read through the listing, relate each line back to the steps noted under Stage 2 on page 57.

Stage 3

Coding

```
100  REM WATER BED PROBLEM
110  REM ASSIGN VALUES TO VARIABLES
120  LET L=6.5
130  LET W=4.5
140  LET H=3
200  REM CALCULATE VOLUME OF BED (C)
210  C=L*W*H
220  REM CALCULATE GALLONS OF WATER (G)
230  G=C*6.25
235  G=INT(G+1)
240  REM CALCULATE WEIGHT OF WATER (P)
250  P=C*62.5
255  P=INT(P+1)
260  REM CONVERT GALLONS TO LITRES (LT)
270  LT=G*4.546
275  LT=INT(LT+1)
280  REM CONVERT POUNDS TO KILOGRAMS (KG)
290  KG=P*.4536
295  KG=INT(KG+1)
300  REM DISPLAY RESULTS
```

This extra line has been inserted to round up the answer to the nearest whole number. This also applies to lines 255, 275, and 295 – see explanation below.)

```
310 PRINT"YOUR BED WILL HOLD ";G;" GALLONS
    OR ";LT;" LITRES"
320 PRINT"THIS WILL WEIGH ";P;" POUNDS OR
    ";KG;" KILOGRAMS"
330 END
```

NOTE

The colon (:) allows placing of more than one statement on a single program line.

The semi-colon (;) is used in program statements to allow several printed sections to be joined together onto one line.

Look at lines 235, 255, 275 and 295. They have been placed to round up the answer to the nearest whole number. Without this, the answers, while being accurate, would extend to several decimal places.

Suppose that Joe's floor will hold a maximum of only 2,240 pounds or 1,000 kilograms, and the weight of the bed works out greater than this. A line could be added to read:

```
325 IF P> 2240 OR KG > 1000 THEN PRINT"SORRY JOE,
    YOU CAN'T HAVE THE BED, IT'S TOO HEAVY!"
```

(The right-facing arrows (>) in the program mean "greater than".)

This program could be adapted to enable you to enter different values to the variables and enter different maximum weights. In other words, you could adapt it to show various loads on areas or on things such as trucks, railway wagons, or whatever you like.

From this simple example it can be seen that computers can be particularly useful for carrying out repetitive calculations, especially in situations where complex formulae are used, such as in the analysis of research data or laboratory experiments.

There are many other things that could be done, but this should give you a good idea about how to design and code a program. The actual commands used may vary slightly between different makes of computer.

CODING A PROGRAM WITH WORDS

IMPORTANT NOTE

*The word PRINT as a single statement has the same effect as a carriage return on a typewriter. In effect, it inserts a blank line and on screen makes the message easier to read. In the program it represents lines of text in double spacing.

So much for finding out about numbers, but what about words? Here's a program that gives results in words and could be used for a children's birthday party. The intention of the program is for each child to enter his or her name and receive a message. The child who is having the party – we'll call him Jim – is to receive a special message.

In this program we must insert the dollar sign $ to indicate that we are using string variables (see page 27) instead of numbers.

```
100  INPUT"WHAT IS YOUR NAME";A$
```

(The child's name is input)

```
110  IF A$="JIM" THEN GOTO 200
```

(This tells the computer to go to line 200 if the name is Jim)

```
120  PRINT "ENJOY YOURSELF AT THE PARTY";A$
```

(If any other name is inserted line 120 will be printed)

```
*130 PRINT
```

```
140  GOTO 100
```

(The program then returns to the beginning to await input of another name)

```
200  PRINT "HAPPY BIRTHDAY JIM"
```

(Line 200 is printed if the name is Jim)

```
*210 PRINT
```

```
220  GOTO 100
```

(Again the program returns to the beginning to await input of another name)

You will see that this program keeps looping back to the beginning as soon as it has been run.

★ ★ ★

These examples are not intended to teach you programming, but rather to give you an idea what programming is all about. If you think programming could be fun your computer-operating manual will set you on the right course for your particular machine, and there are also many good books available dealing with the subject of programming in BASIC.

Make sure, if you are buying from a mail-order catalogue or by some other indirect means, that the title of the book on programming includes the word BASIC. Some books on "microcomputer programming" are about machine language, rather than BASIC, which is what you should look for.

Other good sources of information on programming are computer magazines. Many are aimed at the user rather than the technician and you will learn a lot by studying the program listings they publish.

Having said a lot of positive things about programming, it is now time to look at the other side of the coin.

A lot of time is wasted producing programs that are virtually worthless. In other words, the amount of time and effort spent is far out of proportion to what the program achieves.

Good programs can be written on an almost unlimited number of subjects, but the guidelines for doing this are negative rather than positive ones – in fact, they comprise lists of "don'ts".

Don't write a program unless it is for a task you do frequently – more than two or three times a year. It will probably take you at least three or four times longer to program the computer than it would take to do the job manually, so unless the program is used frequently it will be of no ultimate benefit.

Don't write a program unless the task is something you would do more often if it was easier to carry out. In its simplest terms, this is the reason why computing has gained such power.

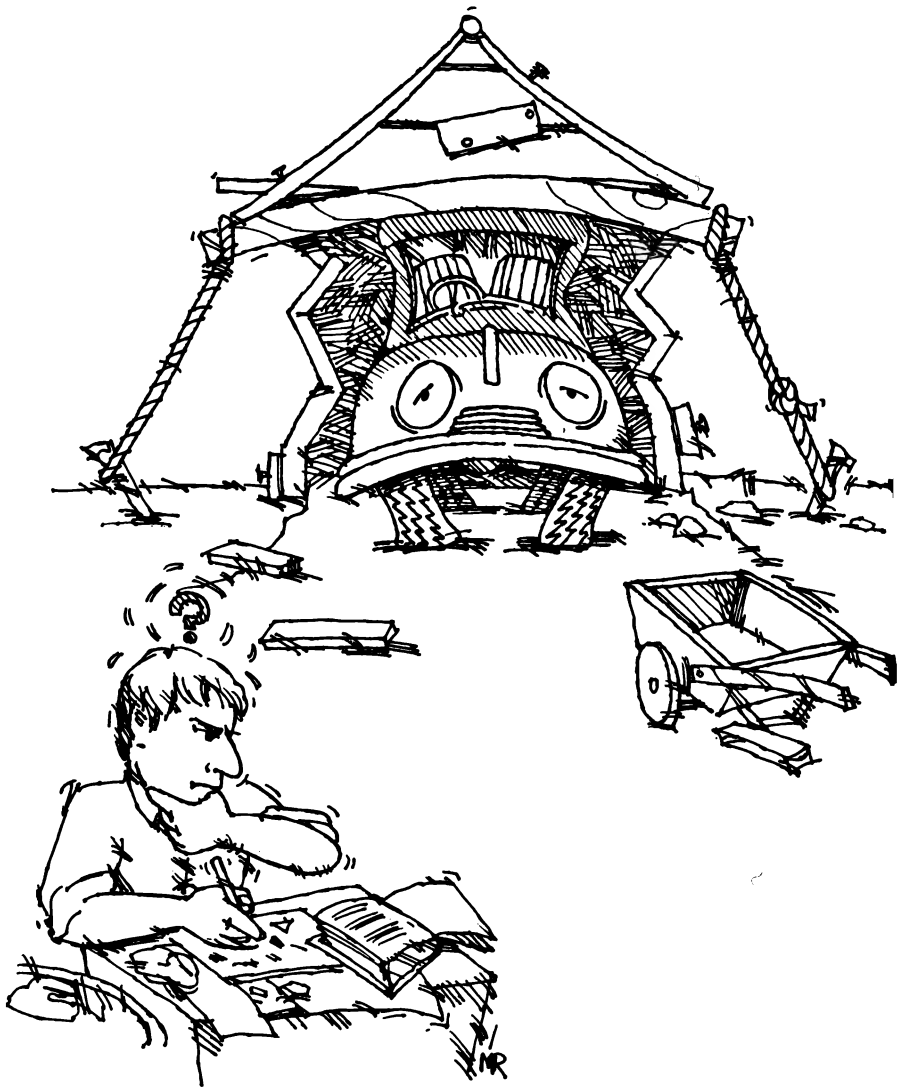
Don't write a program if you could do the task itself in less than five minutes.

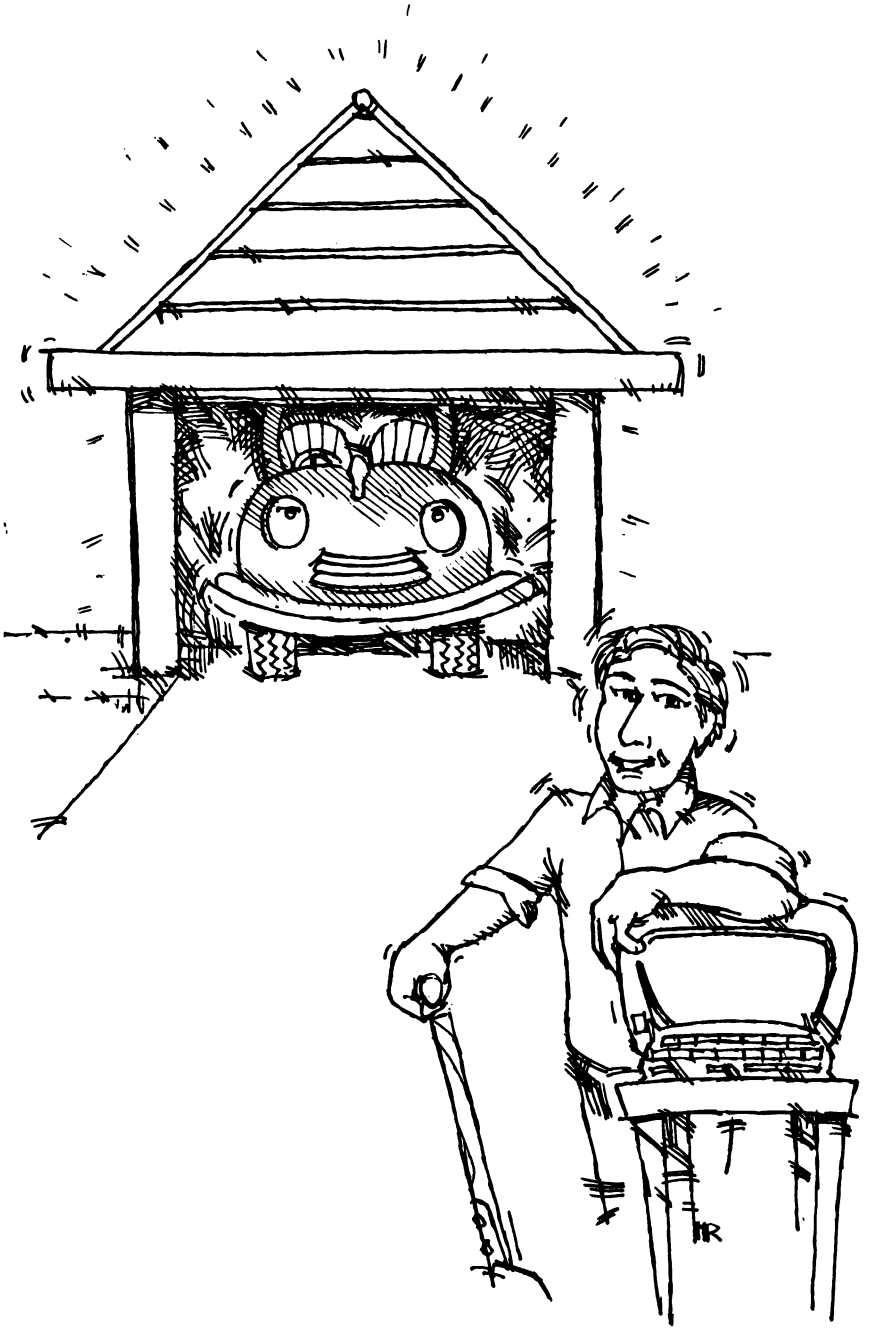
Don't write a program that could be done more easily with a calculator, a small box and some filing or index cards, or a pencil and appointment book.

You can balance your chequebook, update the average personal mailing list and make a note of future appointments faster by hand than computer, and it is much cheaper that way. Commercially available programs used to carry out these tasks have a legitimate place if you have the volume of work to support them. If you do need computer help for such tasks, buy a ready-made program which meets your requirements rather than spend time re-inventing the wheel by writing your own.

Don't write a computer program if accuracy is not essential. The computer can tell that 2.477557733 cans of paint are required, when you would have guessed two and a half or perhaps three. If paint is available only by the can, accuracy to nine decimal places is not going to be of value to you.

All these don'ts consider one major factor – time. If your basic aim is to make good use of your computer and, if the computer is to free your mind for more important things, it must enable you to do the mundane tasks in less time. If a task actually takes





The step-by-step method recommended in programming problem solving by computer helps to ensure accuracy.

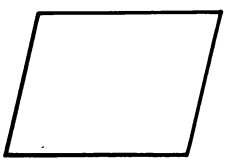
longer with a computer than without, your machine has become a hindrance rather than an aid.

FLOWCHARTING

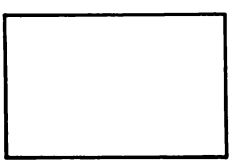
In the initial design phase of a program it is possible to lay each step out in a graphic form to give a better idea of what the program is going to do. This is known as "flowcharting".

Flowcharting is, in effect, a visual map of a program showing how it will be run by the computer. It is an excellent way of checking the program before coding it into the machine.

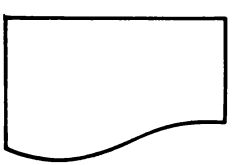
The flowchart consists of a number of standard symbols with a particular meaning joined together by lines and arrows showing the direction of the "program flow". Although there are more than 30 symbols available, you may now concern yourself with only five of them.



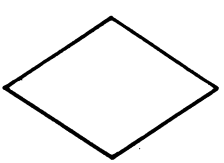
INPUT refers to any information entered through the keyboard.



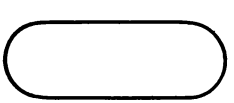
PROCESS means any processing by the computer of information entered through the keyboard or generated within the program during its run.



PRINT means any output to the screen or to a printer.



DECISION means a point in the program where information is tested for alternative options. This is usually a simple "yes" or "no".

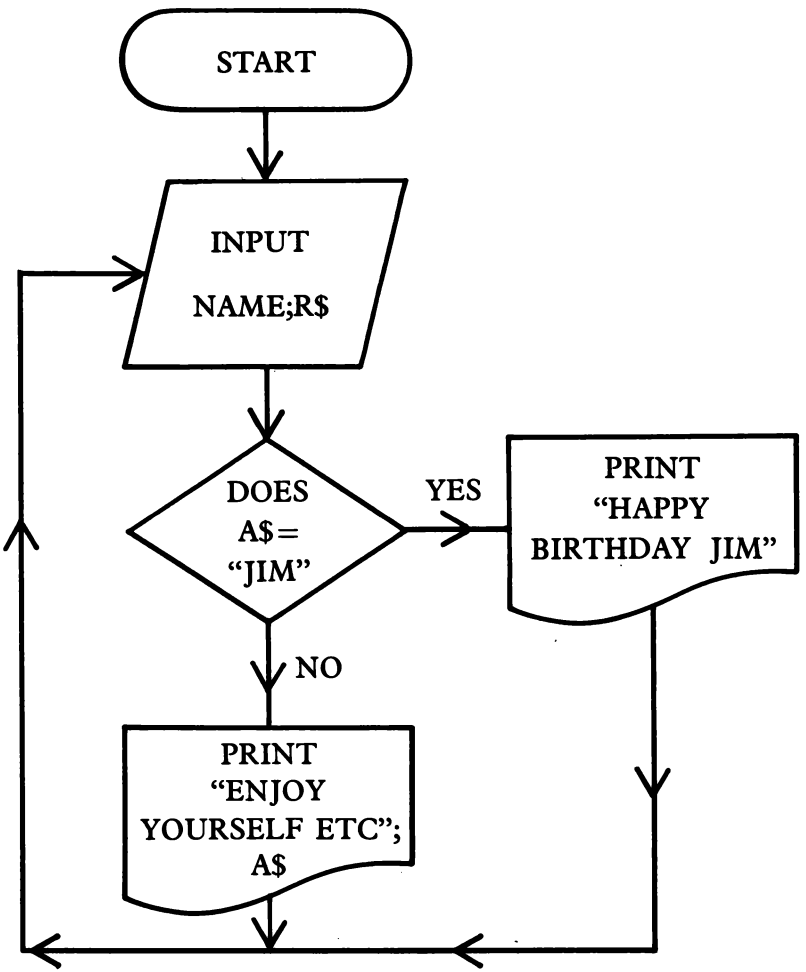


Correctly known as an INTERRUPT, this symbol is normally used to mark the beginning and end of the program.

A flowchart should be constructed early in the program-design phase, just after each stage of the program is written down, together with the formula needed for each calculation.

Flowcharting at this stage will enable any changes to be worked out if the program flow presents any problems. A flowchart does not need to be very detailed, but it should contain enough information to show what is taking place.

Now let us flowchart the program that gives the birthday party messages (see page 60).

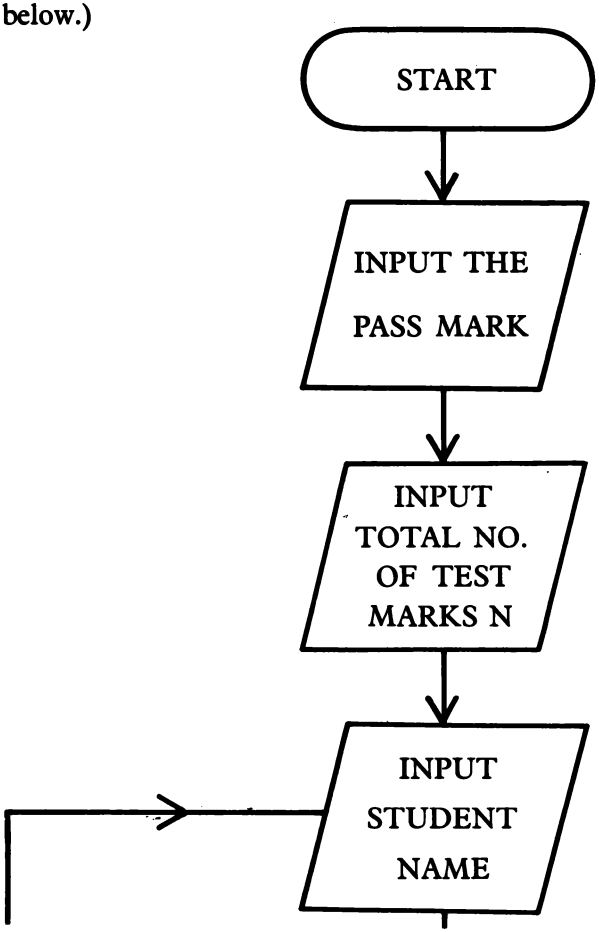


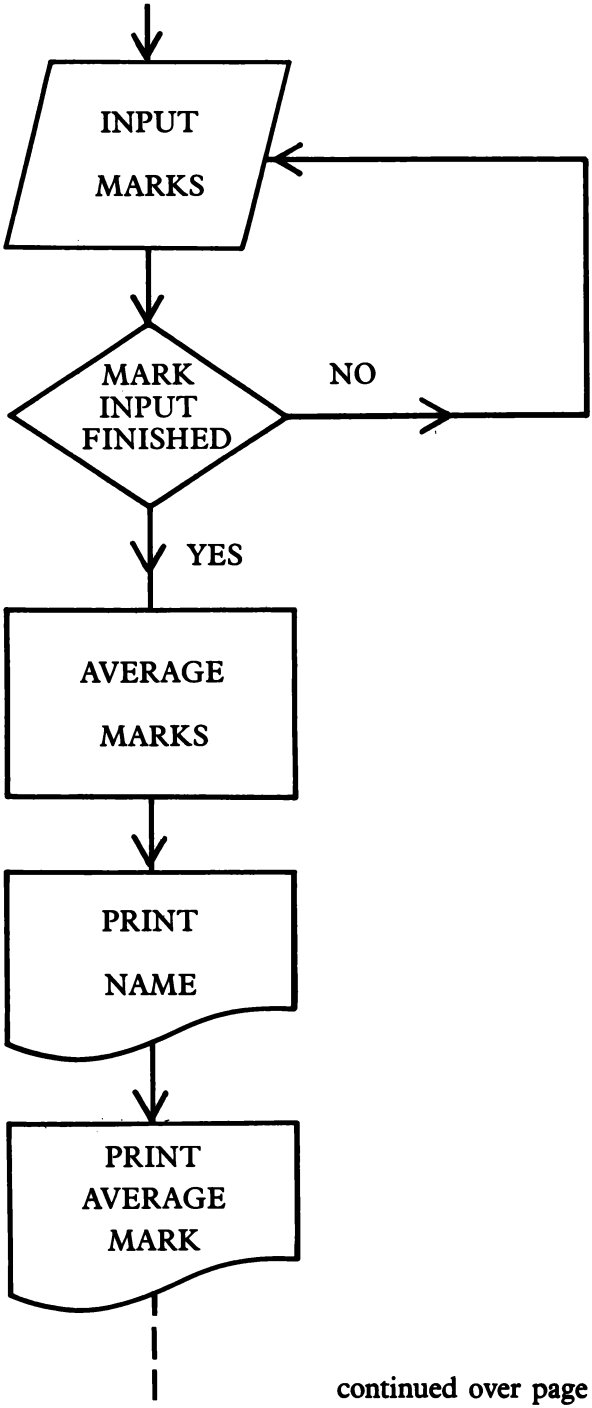
A look at the flowchart reveals that the program will start and then go into a continuous loop until it is interrupted or the computer is turned off.

When a name is entered, the program compares it to see if it is "Jim". If so, it branches out for the special birthday message. If not, it ignores the special message and prints the other one instead and then goes back and waits for the next name.

The final flowchart shows a program designed for a teacher who has test papers to mark. It is a large class and there are about 15 test papers to mark for each student. The program has to average the marks of each student and then print out the average mark together with the student's name.

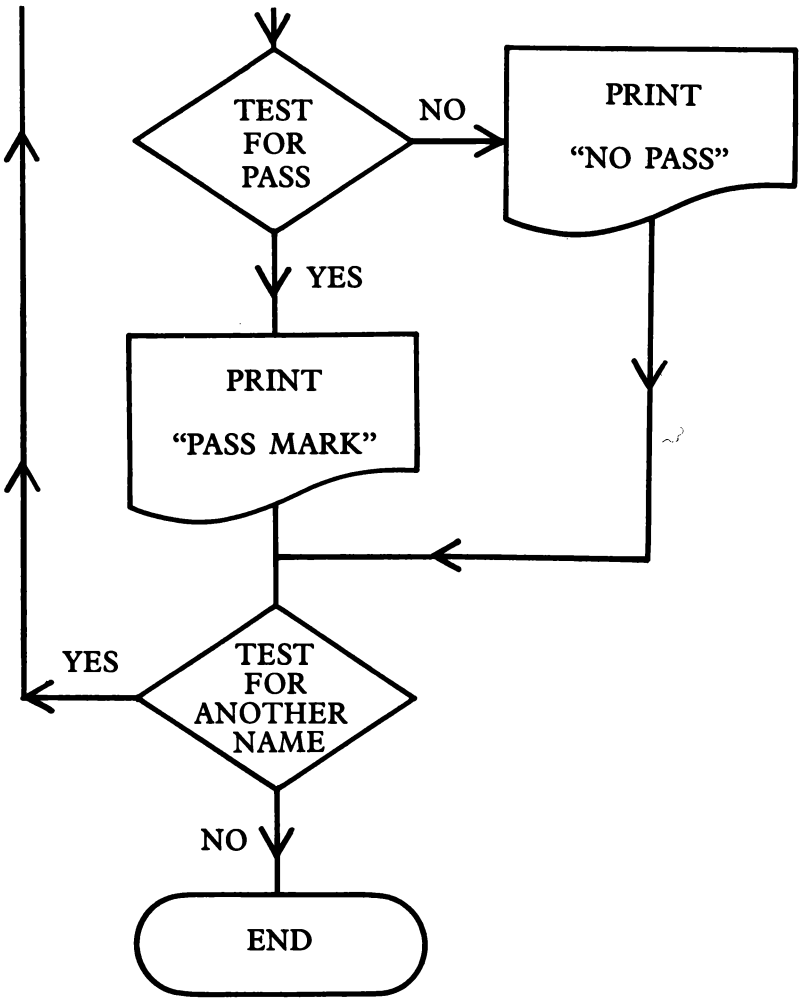
Having printed out the student's name and average mark, the program will also print out a comment to indicate whether the student has passed or failed the series of tests. (See explanation below.)





continued over page

continued from previous page



Once the pass mark has been input, the total number of marks to be processed is also entered. This is assigned to the variable N. Each time a student's mark is entered the program will check to see if this is the last one. If not, it then loops back and waits for the next mark to be entered. Once the number of inputs equals the number assigned to the variable N, the program moves to the next stage.

When the marks have been averaged the program compares them with the original pass mark. If the average is equal to or

higher than the pass mark, the message "pass mark" is printed with the students name and average mark. If the average is below the pass mark, "no pass" is printed.

★ ★ ★

You will see from these few examples that the flowchart is a valuable aid to program design. While trying to translate a coded program from one machine to another can be a complicated job, having a good flowchart for that program can sometimes be far more helpful.

SUBROUTINES

In a program there will sometimes be a form of calculation that must be used several times. To save re-typing these lines over and over again "subroutines" should be used.

Subroutines can be grouped at the beginning or end of a program, and the statement GOSUB (a BASIC language command) sends the program to them each time they are needed. At the end of the subroutine the command RETURN will send the program back to the next statement at the point where it branched off.

A subroutine could be used to check the student's marks against the pass mark. To do this an "IF ... THEN" test would be applied. Another sign, ">=", meaning "greater than or equal to" would also be used to compare the student's mark against the required pass mark. (See explanation beneath the following.)

Part of the program could look like this:

```
100  .....
200  .....
250  REM. AVERAGE MARKS
260  .....
270  GOSUB 500          (The subroutine appears at line 500.)
280  REM. CHECK FOR MORE NAMES
290  .....
300  END
....
....
500  IF AV >=65 THEN PRINT"PASS MARK":RETURN
```

In the subroutine the program tests the average mark against the pass mark which has been set at 65. If it finds that the student's average is equal to or greater than 65, the message is printed out. If the average mark is less than 65 the program will print

out the message "NO PASS" (as in the flowchart). The program will then return to line number 280 and continue.

Subroutines are very useful and can save a lot of memory space. It is possible to write programs that consist of a small "control program" and a number of subroutines. Many programmers assemble libraries of subroutines which can be used in various types of programs – this saves rewriting them each time they are needed.

★ ★ ★

The foregoing is a brief look at how programs are designed and written, and the examples will give you some idea of what programming is all about.

A final point: when you write a computer program containing established formulae or conversion tables, check with an example you know is correct and use it as a model to make sure your program comes up with the correct answer. Once you know it is running correctly you will be able to rely on the results you get.

Nobody – and that means even you – writes programs that are entirely free from errors every time. Every program must be checked and tested. Some mistakes may be obvious, such as being asked to add a negative number of eggs to a recipe, but other errors are much harder to find and often disguise themselves as legitimate answers.

There are many worthwhile things to program. Don't waste time writing irrelevant programs. Make your computer function as it was intended – to provide you with time for bigger and better things.

Chapter 8

Programs look like this

Having looked at the general outline of programming, we shall now look at some actual program listings and see how they work. However, it is important that you remember that program listings differ with various computer systems and it is essential to obtain a manual for a particular system if you intend to write programs for it.

Although programming in BASIC is standard to a point, the variations each manufacturer has introduced could trip you up. You will find out what these are in the various manuals that come with each computer. Use of these manuals is the first step in programming. When you are familiar with them you can progress to other books on programming your particular type of machine, or you may be expert enough to rewrite other programmes so they can be used on your machine.

IMPORTANT NOTE

Before you look through the accompanying program listing there are several things that need explaining. Some of these are commands to the computer; others are part of the computing process.

HOME is a special command to clear the screen. Other computers may use **CLR** or **CLS** for the same purpose.

VTAB and **HTAB** are commands to insert spacing, as with the tabulator or carriage return on a typewriter.

VTAB 3 means go down three lines from the top of the screen.

HTAB 5 means go across five columns.

The word **PRINT** has a similar effect, inserting a blank line.

REM indicates a remark to show what the next section of the program will do. The computer will ignore this line. It is for the user's information only.

INT means integer. In the accompanying example it, and the

rest of the formula noted (M*100+.5), causes the return of a whole number (1, 2, 3) plus two decimal places. This will appear as dollars and cents.

Also note the following:
The dollar signs (for example in the statement PRINT "WEEKLY SALARY=$";W:PRINT) are included within the string variable as part of the permanent line to be printed out. This is done because the dollar sign will always be part of the answer in this program (it concerns salaries!), while the amount beside the dollar sign will vary with each run of calculations.
See also the explanation of the punctuation used on page 59.

EXAMPLE: SALARY ANALYSIS PROGRAM

A visitor once complained to me that in these days of rapid inflation his salary seemed to keep going up, but he was not sure how much per hour his time was worth. A few minutes work on a calculator solved the problem, but he kept glancing at the computer in the corner of the room and finally said, "Let's see you work it out on that thing."

This program, called Salary Analysis, takes an annual salary and breaks it down into monthly, weekly and hourly rates. No allowances have been made for tax, as this varies from person to person. (The program was written for the Apple II+ computer which allows multiple statements on one line. Some computers can take only one statement on each line).

Read the explanation beneath the program in conjunction with the program listing.

```
100  HOME
110  VTAB3:HTAB5:PRINT"SALARY  ANALYSIS"
120  REM.INPUT  DATA
130  VTAB10: INPUT"ENTER YOUR ANNUAL SALARY
     $";A:HOME
140  REM.CALCULATE
150  Y=A
160  M=Y/12:REM.MONTHLY AMOUNT
170  M1=INT(M*100+.5)/100
180  W=Y/52:REM.WEEKLY AMOUNT
190  W1=INT(W*100+.5)/100
200  D=W/5:REM.DAILY (5 DAY WORKING WEEK)
210  D1=INT(D*100+.5)/100
220  H=W/40:REM.HOURLY (40 HOUR WEEK)
230  H1=INT(H*100+.5)/100
```

```
240  REM.OUTPUT
300  VTAB3:PRINT"ANNUAL SALARY=$";Y:PRINT
310  PRINT"MONTHLY SALARY=$";M1:PRINT
320  PRINT"WEEKLY SALARY=$";W1:PRINT
330  PRINT"DAILY RATE=$";D1:PRINT
340  PRINT"HOURLY RATE=$";H1:PRINT
350  REM.END PROGRAM OR RE-START
400  INPUT"ANOTHER RUN (Y/N) ?";A$
410  IF A$="Y" THEN GOTO 100
420  HOME:VTAB5;HTAB5:PRINT"<<PROGRAM
     ENDED>>"
430  END
500  REM.COPYRIGHT 1981
510  REM.THE ELECTRIC APPLE MAGAZINE
```

As you can probably see, this is really just an exercise in progressive division, but the above shows how it needs to be written for the computer.

Lines 100 and 110 clear the screen and then display the title of the program. Line 120 reminds the programmer to input the data. Line 130 inserts some extra spacing and requests the input of the annual salary ($";A). The word HOME clears the screen ready for calculations which are made in lines 150–230. The alternative lines (170, 190, 210 & 230) with the calculation enclosed in brackets are used to convert the answer to a whole sum of dollars, plus two decimal places to give the cents. Output to the screen (or results) starts at line 300 and then at line 400 the user is asked if the program is to be ended or another run of calculations made. (In line 400 Y/N means "yes" or "no".) If the answer is "yes" the program goes back to line 100 and starts again.

If the answer is "no" the program flow continues, the screen is cleared and the words "program ended" appear.

EXAMPLE: FOREIGN CURRENCY CONVERTER PROGRAM

If you deal with foreign currencies it is important to know the various rates of exchange. Perhaps you intend going on an overseas trip. By entering the amount of money you intend to take, or perhaps the price of something you intend to buy, you will be able to find out the true cost when conversion rates are taken into account. You could also use the following program in conjunction with shipping documents, calculating export costs or many similar applications.

Look quickly through the program listing here, then refer to the detailed explanation below for a step-by-step outline of how it works.

```
10000 CALL-936
10010 VTAB8:HTAB6:PRINT "FOREIGN CURRENCY
      CONVERTER":PRINT:PRINT TAB(6) "$$$$$$$
      $$$$$$$$ $$$$$$$$$"
10020 FOR P=1 TO 1500:NEXT P:HOME
10030 VTAB3:PRINT "THIS PROGRAM MAKES QUICK
      CONVERSIONS": PRINT:PRINT "FROM
      DOLLARS TO FOREIGN CURRENCIES":PRINT:
      PRINT "AND FROM FOREIGN CURRENCIES TO
      DOLLARS":PRINT:PRINT
10040 PRINT:PRINT:PRINT "<< PRESS ANY KEY TO
      CONTINUE>>":GET A$:HOME
10050 CALL-936
10060 VTAB4:INPUT "WHAT IS THE EXCHANGE
      RATE?";R:PRINT:PRINT:PRINT
10070 INPUT "NAME OF FOREIGN CURRENCY?";T$:
      PRINT : PRINT
10080 HOME
10090 VTAB5:PRINT "DO YOU WANT:"
10100 PRINT:PRINT "(1) DOLLARS TO ";T$:PRINT:
      PRINT
10110 HTAB 15:PRINT "OR":PRINT:PRINT
10120 PRINT "(2) ";T$;" TO DOLLARS"
10130 PRINT:PRINT:PRINT:
10140 PRINT:INPUT "ENTER 1 OR 2 ";C
10150 HOME
10160 IF C=1 THEN 10190
10170 IF C=2 THEN 10280
10180 IF C<2 THEN HOME : GOTO 10090
10190 VTAB5:PRINT "WHAT AMOUNT IN DOLLARS
      HAVE YOU":PRINT
10200 PRINT "TO CHANGE INTO ";T$
10210 PRINT:PRINT:PRINT:INPUT D
10220 HOME
10230 H=D*R
10240 F=INT(H*10000+.5)/100
10250 HOME
10260 VTAB5:HTAB10:PRINT D;" DOLLARS=";F;" ";T$
10270 GOTO 10360
```

```
10280 VTAB5:PRINT "WHAT AMOUNT OF ";T$: PRINT
10290 PRINT "HAVE YOU TO CHANGE INTO
      DOLLARS?":PRINT
10300 INPUT C
10310 HOME
10320 H=C/R
10330 F=INT(H*10000+.5)/100
10340 HOME
10350 VTAB5:HTAB10:PRINT C;" ";T$;"=";F;
      " DOLLARS"
10360 VTAB20:INPUT "(1) MORE CONVERSIONS. (2)
      END PROGRAM.";C
10370 IF C=1 THEN 10050
10380 HOME
10390 VTAB10:HTAB10:PRINT "END"
10400 END
10410 REM.
10420 REM.
10430 REM.FOREIGN CURRENCY
10440 REM.CONVERTER.
10450 REM.
10460 REM.B.D.STRONG
10470 REM.
10480 REM.COPYRIGHT OCT. 1980
10490 REM.ALL RIGHTS RESERVED
10500 REM.
```

First, you have probably noticed that the Foreign Currency Converter uses very high line numbers. It could just as easily start with 10 or 100 if the program is to be used by itself. However, it is possible with most microcomputers available to turn a program into what is known as a "text file". This makes it possible to "append" or add the program (text file) to the end of another program you may be using currently in the machine. (You will find explanations of how to do this in your systems operating manual. The actual method to be used varies from machine to machine.)

As already stated before the program listing, this currency converting program could be used in conjunction with a program that calculates export costs. However, it is essential that if a text file is to be added onto a file it does not duplicate the original line numbers, or chaos will result and the whole program will become useless. To guard against this it is usual to give text files

that may be appended to another program high line numbers so the programs do not overlap.

Now, for the step-by-step explanation of the program listing.

The command in the first line (CALL -936) means the same as HOME (as explained on page 71). The commands VTAB and HTAB mean vertical and horizontal tab spacing, the same as on a typewriter. They are used to determine the format of the screen output.

In line 10010 you will see the title of the program and also a row of dollar signs. HTAB6 puts the title six spaces across the screen. PRINT adds a blank line, and then PRINT TAB(6) underlines the heading with the dollar signs. HTAB6 and PRINT TAB(6) do the same thing on the machine I use. Which method you use is a matter of preference.

Line 10020 holds the title on the screen long enough for it to be read and then replaces it with line 10030. This is known as a "loop", which in this case merely counts from 1 to 1,500 before erasing the screen (HOME) and continuing with the next instruction (line 10030).

Line 10030 then tells you what the program will do. The extra PRINT statements (inserting blank lines) mean that text on the screen will be double spaced, much neater and easier to read than single-spaced text.

Line 10040 inserts three blank lines after the text and then tells you that if you "press any key" in the keyboard the program will continue to the next stage. Two instructions on the end of this line – GET A$ and HOME – are commands used by the Apple computer. If you were using the TRS-80 computer the commands would be GET INKEY$ and CLEAR. These commands are used simply to halt the program until another key is pressed and then clear the screen for the next lot of text to be printed.

The next part of the program asks for the exchange rate. This can usually be found on the financial pages of a newspaper or it can be obtained from any bank. The next item required is the name of the foreign currency. Use the proper one, or make one up. Whatever you do it will be used again in the program. In the case of this program the currency has been assigned the variable T$.

If, for example, you input the name francs, the next question (lines 10090–10120) will ask whether you want a conversion from dollars to francs or francs to dollars.

Line 10140 asks the user to enter the figures 1 or 2 into the computer. Already on the screen will be the following display:

DO YOU WANT
(1) DOLLARS TO X
OR
(2) X to DOLLARS
ENTER 1 OR 2

X represents the name of the foreign currency already entered. This entry is then presented within the program as the variable C.

The program then looks at what you have entered with lines 10160 and 10170. If you have accidentally entered a number greater than 2 (>2), line 10180 will clear the screen and return the program to line 10090 and will keep doing so until either 1 or 2 is entered. This is known as an "error trap".

If you have entered the figure 1 the program will jump the next couple of lines to line 10190 and will continue through that part of the program (making the appropriate conversions of dollars to T$ and rounding them up to two decimal places) until line 10270, where it jumps on to line 10360. If you entered the figure 2 the program will jump to line 10280, to the other set of calculations (from T$ to dollars) and also ends up at line 10360.

Line 10360 gives the option of more conversions or ending the program. Once again it asks for an entry of 1 or 2, but this time there's a difference. In line 10370 it checks to see whether 1 has been entered. If so, the program returns to line 10050 where the screen is cleared ready for more conversions. If anything other than 1 has been entered the program will continue, clearing the screen in line 10380 and printing END in the middle of the screen in line 10390. Line 10400 tells the computer where the program actually ends.

Lines 10410–10500 are REMarks, not actioned by the computer, but used to include information about the program for the user, such as title, author and any copyright information or mailing address. This information written against REMs will not print on the screen, but can be seen when the actual program is listed or printed out on your printer.

With this program there are a few other points you may wonder about, so let's deal with them here as well.

In lines 10190 and 10200 a sentence is broken between two statements. Why? The simple answer is that it would not fit across the screen in one line. This, and the insertion of blank lines (PRINT), has to do with what is known as "formatting" or "layout". Good layout makes the program easier to read and

understand – a very important consideration for those not used to computer operation. It also looks much neater than a jumble of single-spaced lines filling the whole screen.

In lines 10230 and 10240 you will see the "variables" H and F. I used H and F because they hadn't already been used. Variables can be anything you choose because they concern only the programmer; the user is concerned only with what appears on the screen when the program is running. Whole words can be used as variables if you like, but this makes for messy and confused coding and also uses up extra memory space in the machine.

EXAMPLE: NAME OF PRODUCT

Computer programs are not totally involved with numbers. The next program makes up names, in this case for some new product about to be marketed. (One word of warning about this type of program – sometimes it will result in rude or obscene words purely as a result of random letter combinations.)

Again, look quickly at the program listing, then read the lines in conjunction with the explanation at the end of the program. You will notice that this is a different style of programming, with only one statement to each line.

```
1100  HOME
1150  TAB 10
1200  PRINT"THE NAME GAME"
1250  PRINT
1300  PRINT"YOU ARE AN ADVERTISING EXECUTIVE"
1350  PRINT"WITH A NEW PRODUCT TO NAME"
1400  PRINT
1450  PRINT"TYPE IN TWO LETTERS SUCH"
1500  PRINT"AS DR GL SH ETC"
1550  PRINT"AND THEN PRESS 'RETURN' ",A$
1600  INPUT A$
1650  PRINT
1700  FOR I=1 TO 5
1750  FOR J=1 TO 5
1775  PRINT A$
1800  IF I=1 THEN GOTO 2050
1850  IF I=2 THEN GOTO 2150
1900  IF I=3 THEN GOTO 2250
1950  IF I=4 THEN GOTO 2350
2000  IF I=5 THEN GOTO 2450
2050  PRINT "A";
2100  GOTO 2500
```

```
2150  PRINT "E";
2200  GOTO 2500
2250  PRINT "I";
2300  GOTO 2500
2350  PRINT "O";
2400  GOTO 2500
2450  PRINT "U";
2500  IF J=1 THEN GOTO 2750
2550  IF J=2 THEN GOTO 2850
2600  IF J=3 THEN GOTO 2950
2650  IF J=4 THEN GOTO 3050
2700  IF J=5 THEN GOTO 3150
2750  PRINT "S",
2800  GOTO 3200
2850  PRINT "P"
2900  GOTO 3200
2950  PRINT "T",
3000  GOTO 3200
3050  PRINT "R",
3100  GOTO 3200
3150  PRINT "B",
3200  NEXT J
3250  NEXT I
3300  PRINT
3350  PRINT"THAT'S ALL FROM YOUR ENTRY."
3400  PRINT
3500  PRINT"TRY AGAIN (Y/N)?",B$
3550  INPUT B$
3600  IF B$="Y" THEN GOTO 1300
3650  IF B$="N" THEN GOTO 3750
3700  GOTO 3400
3750  PRINT"PROGRAM ENDED"
3800  END
```

The Name Game is a bit of fun, which prints out combinations of letters you have entered, together with a list of vowels (lines 2050–2450) and consonants (lines 2750–3150). This is done by using "loops" which pick up the letters in turn and add them together to make a word, although many of the resulting "words" won't make sense. The loops do all the work. In a nutshell, your two letters are printed, loop I adds a vowel and loop J a final letter. This sequence is repeated five times by each loop, ending up with five rows of words each five lines deep – a total of 25 words on each program run.

Now for a more detailed explanation for those who want to understand the "nitty gritty". (Feel free to skip this explanation if you wish – if all you require at this stage is knowledge of how a program looks, the above will be sufficient.)

Lines 1450 and 1500 ask you to enter two letters. Your input will be stored as the variable A$ in line 1600 (Remember, the dollar sign denotes a string variable using letters, not numbers.) Lines 1700 and 1750 show the two loops, one called I and the other J.

When this program is run, your letters (variable A$) are printed by line 1775. We'll assume that you have started with the letters DR. Every time the program passes through line 1775 it will print these letters.

On the first cycle (I = 1) line 1800 comes into effect and sends the program to line 2050 where it picks up the letter A. So far we have the original letters DR with the addition of A. Then the program moves to line 2100 and is sent to line 2500 where it checks the value of J (= 1). Then it is sent to line 2750 where the letter S is picked up. Thus the first "word" (DRAS) is completed.

The second word in the first row is formed by adding a different consonant to the first three letters already noted (DRA). Immediately after the first word is completed (at line 2750) the program is told to progress to line 3200, which tells it to pick up the next letter from J loop. It finds the location of this letter (line 2850) by looking at line 2550 (J = 2). The letter is P, giving us the second "word" (DRAP).

DRAP and DRAS are, of course, not real words, but on the third and fifth cycles of loop J in this sequence, the words DRAT and DRAB will be formed.

After the completion of each word, the program checks whether the cycle of five values for each loop has been completed by referring to lines 1700 and 1750. If either I or J is greater than 5 the program would progress through to line 3350 without actioning any of the previous lines. (Line 3350 lets the user know the run has been completed.)

Then it reads line 3500 which asks if you would like to try again. (The Y/N is simply a shortened way of saying "YES or NO"; your answer is given the variable B$) and the program progresses through to lines 3600 or 3650, depending on whether the answer was positive or negative. If positive (Y) the program would go back to line 1300 and start again. If negative (N) it would jump over line 3700 to line 3750 and print PROGRAM ENDED.

Line 3700 shows another way of error trapping. If neither Y nor N is entered the program will jump back to line 3400, insert a blank line and then go to line 3500. It will keep doing this until either Y or N is entered. Without it an inexperienced user may enter a wrong letter and then sit helplessly while the program does nothing or "hangs" forever.

These programs show in more detail than those in Chapter 7 how a computer goes about its work. You will also see that if you wish to write programs you must try and analyse your problem thoroughly before and during the design stage.

There are many aspects of programming that have not been covered because my intention has been to familiarise you with the process rather than teach you how to program. That is best done by consulting your computer manual and one of the many specialised books available.

Chapter 9

Choosing your micro and software

Before you go rushing out to buy a microcomputer there are some questions that must be settled.

What do you intend to do with it?
How much can you afford to spend?
How much do you actually know about the various makes and systems available?
How much software is available?
Is the right type of software available for the particular task you have in mind?

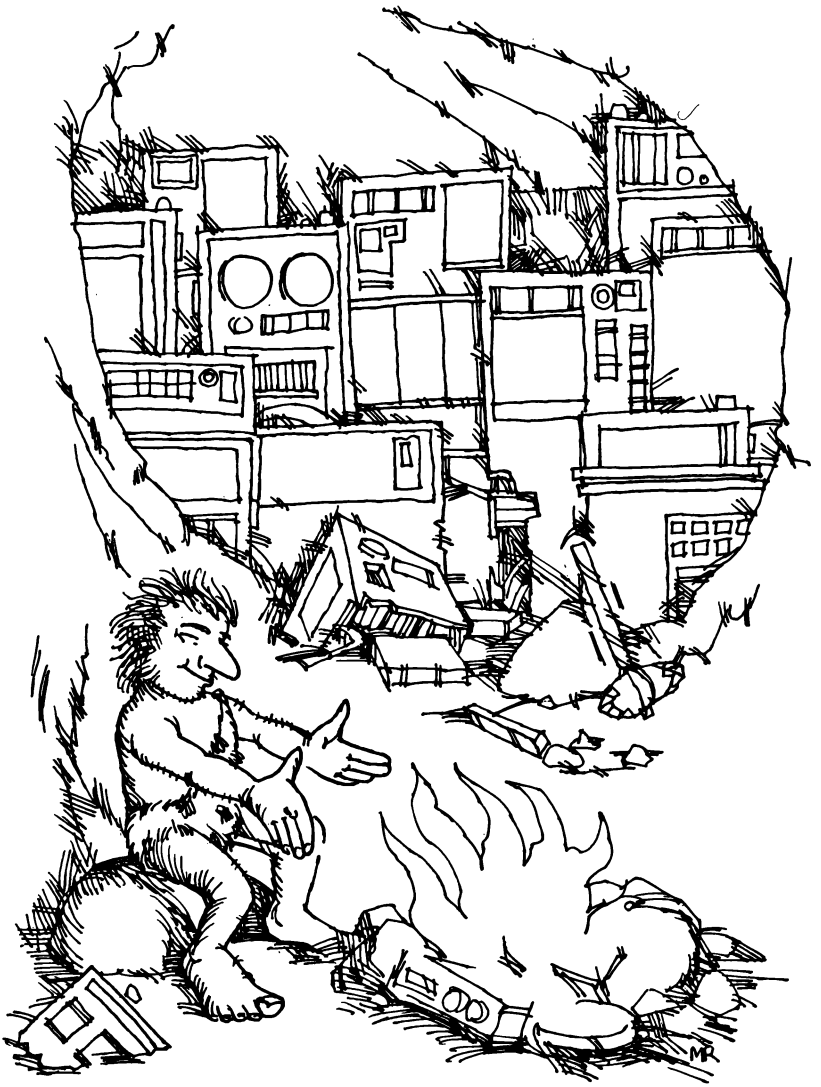
If you are buying a microcomputer just for personal and recreational use, the above may not concern you too much. The only problem on your mind will be how much you can afford to spend. However, apart from this there are still a number of points to be considered which apply equally to buying a computer for business or pleasure.

DO YOU REALLY NEED A COMPUTER AT ALL?

This is a hard question to face honestly, but if you have to compile only a few sets of figures, balance the family bank statement or work out some simple production costs, forget the computer. Go out and buy a good pocket calculator. It will be much easier and cheaper to operate and will do the job just as well.

Perhaps you have a range of tasks which call for something more than a simple pocket calculator, but are not sufficiently difficult to require solving by a computer. In this case take a good look at the range of programmable calculators available. Some have both plug-in software cartridges which cover a wide range of applications and enough memory space to carry out fairly long and complex operations.

If any of these meet your requirements purchasing a computer would be like driving to work in a Rolls Royce when a bicycle



First, define your needs....

would do. In other words, it would be an expensive luxury.

Is your business so large that you have to do all your own accounting? Many accountants offer a management accounting service for small businesses. It could be a lot cheaper to use their computer instead of making your own purchase.

Computers are sometimes purchased by people who see them as infallible electronic oracles.

"If we had a computer to handle it all, then everything would be all right," goes the cry.

The truth is that if a business operates on management by confusion, without proper systems and planning, a computer is the very last thing that is needed. It will only make things worse. The only advantage would be that bad managers would have something to blame instead of themselves and their sloppy methods.

Always remember that computers do not make mistakes – but people do! If incorrect data finds its way into the computer, someone has been responsible for preparing it or entering it in that way. If a present system is full of errors the system needs changing before any computer processing is capable of streamlining it.

You will be told time and time again that computers can store vast amounts of information which they process at high speed. One point that is often overlooked is that information must first be entered into the machine.

Successful operation of a computer means a commitment of time. If you have one, two, three or more years' worth of records to store on computer, it will go in only as fast as you are capable of operating the keyboard. If you are a two-finger typist, obviously this could take some time. Records may also have to be updated on a daily, weekly or monthly basis. This all takes time and once commenced must be continued for those records to be of value.

Remember when you set out to key in all those records that keeping them up to date must become as much a habit as brushing your teeth every day. What seems to be a great idea may founder and sink once the novelty has worn off.

IF THE ANSWER IS "YES", WHERE SHOULD YOU START?

If you have thought of all these things and decided that a microcomputer would be an advantage, the next step is to find out what is available. Be prepared to spend some time gathering information about various systems. Check the capabilities of those that fall within the price range you can afford.

Never forget that the basic job of promotional literature and sales staff is to sell the product, if possible from the top of the range with all the "bells and whistles" that go with it. Also, to be fair, unless you ask enough questions sales staff may assume that you know much more than you actually do.

In one documented case a businessman called at a computer

showroom, asked for a quick demonstration and then left with a system. The very next day he was on the phone screaming like a stuck pig. The system wouldn't work; nothing would happen. He had been robbed!

After checking that the system was actually plugged in and the various switches were on, the salesman asked, "What sort of software are you using?"

"Software?" came the reply. "Software, what's software?"

Visit computer showrooms and ask the dealer to give you a full demonstration of various makes and their capabilities. This will usually be quite impressive as there are special demonstration programmes assembled for just such occasions. Once you have got over being dazzled by how clever the machine is, ask for a demonstration of the type of operation you are considering.

Be prepared to ask a lot of questions, even if you think some of them make you sound ignorant. Be honest – you probably know very little about various systems and this is no time for false pride. Remember to ask plenty of "what ifs" and persist to the point of being rude if necessary until you get clear answers you can fully understand. Failure to understand the system you are buying will be little comfort when you are hunched over the keyboard watching garbage stream out for the umpteenth time and wondering how it all happened.

Remember, the final decision is yours and you have only yourself to blame if you find out later that the system cannot handle what you "thought" it could.

Most computers are sold as basic systems with other bits and pieces added later. Make certain of what you are buying and what it will all actually cost you by the time it is installed in your office or home.

If the computer is intended for hobby use or game playing, cassette memory storage may be adequate. If larger applications are being considered, disk drives will be necessary for efficient operation. You may need only one disk drive, but two are better and some business programs will require two. In this case the "control program" operates from one drive and the second drive handles the storage of data. Some systems will handle more than two disk drives, but do you really need more?

Are you going to use graphics? There are varying qualities of graphics available. How good do yours have to be? Are you content with just having your graphics on the screen or do you want to be able to have them printed out? Not all printers can reproduce graphics.

Do you need a printer? Some of the simpler computers cannot

support a printer. With the more sophisticated systems it is usually just a matter of plugging the printer control card into the computer and connecting the printer's cable between the two.

What type of printer do you need? For most applications a simple dot-matrix printer is adequate, but some have better-quality reproduction than others. (See page 32.)

A word processing application would call for a high-quality printer. The printing out of routine data would not.

Make sure the printer will work on your system without any involved procedures that may result in problems later. Some printers and systems do not work well, if at all, with each other. Ask for a demonstration before you buy – it's the safest way.

Printers vary a lot in price, usually governed by their quality of reproduction and the number of other features they have. Some of these include incremental spacing, underlining, super- and sub-scripting, a selection of speeds, graphics capability, double printing, expanded or condensed print and adaptable pin or friction feed, to name a few.

Some systems can be connected to a home television set as a screen or a **VDU** (visual display unit). Other machines already have their own built-in VDU. The VDU is the better choice as it gives a clearer picture and is easier to work with over a long period. Using a VDU will also save arguments when other family members want to watch their favourite television programme.

FUTURE GROWTH

Can the system be expanded at a later date? You may start out with just the computer, screen and one disk drive. As your demands grow you may want to add another disk drive, printer or any of a range of available equipment.

Some systems are limited in the number of additions they can support. Sometimes the addition of a feature will call for a compromise. You may find that one or more of the existing functions cannot be used at the same time. The reason for this is that the "add-on" needs to use part of the available memory area normally used for another function.

MEMORY CAPACITY

When you buy a computer it will be fitted with a certain amount of memory. This can range from 8K to 48K or higher. Make sure that a small amount of memory is capable of being increased as your requirements grow. Many machines are sold with the maximum amount of memory already included. It is a good idea to have as much memory in the machine as you can afford.

BE AWARE OF POSSIBLE FAULTS

Like any piece of complex equipment, a computer is liable to develop faults. For a check on the reliability and other features of a particular system, try asking dealers about opposing brands. More often than not their comments will be highly biased, but as you do the rounds some sort of pattern may emerge. Then take the information back to the dealers and see what they have to say in defence of their product. This is hardly the ultimate test, but it is better than nothing at all.

A better way to check out a system is to find one already in use. Despite the fact that the user may be biased towards the system in use, a visit can provide valuable information on how it works under real conditions. Talk to a number of people, preferably those who don't sell but who know about and use microcomputers.

Keep away from those technically involved with computers unless you also have some technical knowledge. This is not intended as a criticism, but their interest in computers will be completely different from yours. They are more likely to be concerned with the electronics of a system, whereas you are more interested in the applications.

A conversation between two electronics types about how much slower one system was compared to another sounded alarming. A veritable picture of doom emerged until it seemed that such a system could never be sold. Then came the moment of truth. The time difference being debated was measured in microseconds! The average user would have no interest in such esoteric observations. Look for a user who speaks the same language as yourself.

Accept the fact that your system is likely to develop a fault sometime in its life and it will need servicing. Establish how long repairs usually take. For what period are the various components under warranty? What are the terms of the warranty? Is it "floor-to-door" or does it offer something more reasonable than that? Does the dealer have a serviceman on the spot or will the equipment have to be sent away to be repaired? If the latter is the case and if your business depends on computer operation, find out if replacement or hire equipment is available.

How much time is the dealer willing to spend showing you how to operate the system? Just as you would not buy a new car without driving it first, the same should apply to buying a computer. The dealer should not only demonstrate the computer but should also let you spend some time with a simple program

that will allow you to get the feel of operating the machine. If the dealer seems unwilling to do this, find another dealer.

Computers are complex machines and a dealer who tries to sell them on a quick "cash-and-carry" basis is unlikely to be of much use in providing back-up service or assistance. One way of ensuring good back-up and servicing, however, is to select one of the popular established brands of microcomputer.

You will find that each system has its own unique set of features, which should always be compared with your own requirements. These factors should be weighed carefully, together with the availability of a local dealer, servicing and software.

SELECTING SOFTWARE

Software comes either "off the shelf" (provided by companies that specialise in software production) or it can be written for a specific operation. Ready-made or "off-the-shelf" software programs are available on just about every conceivable application and range from simple games, costing just a few dollars, to complex business and technical programs which may have taken years to test and develop. One of the great advantages of this type of software is that it enables people with no programming skills to get on with the business of computer operation with a minimum of fuss.

When you are looking at a particular computer system do take the time to find out how much and what type of software can be readily bought. In general, the more popular the system, the more software available.

With business or technical applications there may be a problem finding a software package that meets your exact job requirements. The description of the program should outline what it can do. Some types of programs, such as database management systems or mailing lists, can be considerably modified to meet user requirements.

If the program is intended for information storage always make sure to check that it is large enough. Some programs have a maximum input that might not cover your requirements.

Make sure that technical programs are written correctly for your application. Check the origin of the program if possible to make sure both you and the programmer are thinking along the same lines. Programs written in one country may not be correct in another. Terminology can also have different meanings in different situations. For instance "survey program" may work out calculations for a land surveyor, analyse questions being asked about a particular product or establish the ratings for a radio or television station.

Despite any shortcomings, ready-made software packages have a lot to offer. Their biggest advantage is providing access to programs for non-programmers, and for program writers they eliminate the need for continually repeating often-used instructions.

Ready-made software is available either from dealers or by mail-order. If you are buying software directly from a dealer, ask for a demonstration so you can see what you are buying, but be reasonable about it. Some packages, such as games, need only a brief check. More complex programs may come with fat

manuals which should be read thoroughly even before the program is loaded. A quick check of the program instructions will show whether there are any special loading instructions. If you don't understand them ask the dealer to show you how it is done.

Mail-order software is like that for any other product. It ranges in quality from very good to very ordinary. Some companies specialise in handling software from a number of sources. In this case you are unlikely to buy a faulty product as their reputation is at stake in a highly competitive market.

There are also many reliable companies which produce and sell their own software by mail-order. These can be found by

looking through computer magazines. Another way of finding out what is available is to ask a computer owner or contact someone who belongs to a computer club. Most computer owners are usually aware of the standard and reputation of various software being offered for sale.

If none of the ready-made software meets your specifications it will have to be specially written for you. While many computer owners write much of their own software, some specially written or "custom" software may best be done by a specialist in computer programming who is experienced in designing, writing and testing programs.

Custom software is expensive and it takes time to produce. The operation for which it is required must be analysed, the program must be designed, checked, coded, checked, tested over a period of time and then modified as necessary until it does exactly as required. If you have a large operation that demands special software, don't expect it to be written in a few days. Depending on the complexity of the task, it is more realistic to allow weeks and even months for development. This is something on which an experienced software producer should be able to advise you.

Any software, simple or complex, should be clearly documented. Ideally a manual or sheet of instructions should enable someone with little or no experience to sit down and run a program successfully.

★ ★ ★

Choosing either software or a microcomputer system involves more than buying the first items you see. Patience and careful selection will pay off in the long-term, ensuring that you end up with an asset instead of a liability.

CHECK THESE BEFORE YOU BUY

★ *For what purposes will you use a computer?*
★ *How much can you afford to spend?*
★ *How much software is available?*
★ *Will the software handle your work?*
★ *Is the software suitable for local conditions?*
★ *Can you spare the time for computer operation?*
★ *Can the system be expanded later?*
★ *What warranty exists?*
★ *What service facilities exist?*
★ *How long would you be without your computer during servicing?*
★ *Is replacement or hire equipment available?*
★ *How much help does the dealer offer?*
★ *Do you really need a computer after all?*

Chapter 10

Looking ahead – some personal views

A button is pressed in an Auckland home and a computer user transmits a file of information to a friend located in Perth. That file is lodged in a "computer bulletin board" system located in the U.S.A. On checking into the system the Perth user is told the file is waiting for him and he retrieves it. Despite its roundabout route, the message has travelled faster than a disk being airmailed from one side of the Tasman Sea to the other.

A farmer with produce ready to sell wants to compare prices on various markets. From the keyboard of his computer he gains access to a **database**, which may be located nearby or perhaps hundreds of kilometres away, and gets the latest figures.

The database is becoming increasingly important in day-to-day activities. This centralised pool of information can be likened to a giant electronic filing cabinet containing information ranging from reference material to marketing information which may be updated hourly as prices and trends come to hand. Usually consisting of a large mainframe computer capable of storing very large files of information, such a database can be accessed by subscribers (similar to joining a library) using their own computers with a MODEM working through the telephone system (see page 35).

Theatre seats, transport timetables, stock-market trends, supermarket prices, travel bookings, educational and recreational programmes, national and international news are all available through a microcomputer keyboard as and when required.

A glimpse into the future? No, this has already happened. The microcomputer is now enabling ordinary people to play a direct role in the world's biggest and most important business – information.

Information is a commodity essential to our daily lives, but information resources have grown to the extent that many old

recording and retrieval systems have become overloaded and broken down. Think of those who have been frustrated because certain information has not yet been catalogued or, more likely, has been put away in a filing cabinet whose exact location has been forgotten.

The pace of life is continually increasing; decisions are made faster and with a greater dependence on up-to-the-minute information. "I want it yesterday," is now the norm rather than the exception. Only computers are capable of sifting, sorting and selection of information sufficiently fast to meet these demands.

Because of its flexibility and widening capabilities, the computer has become a very important tool in today's society; but, like any other major technological advance, it has also been the focus of negative attitudes.

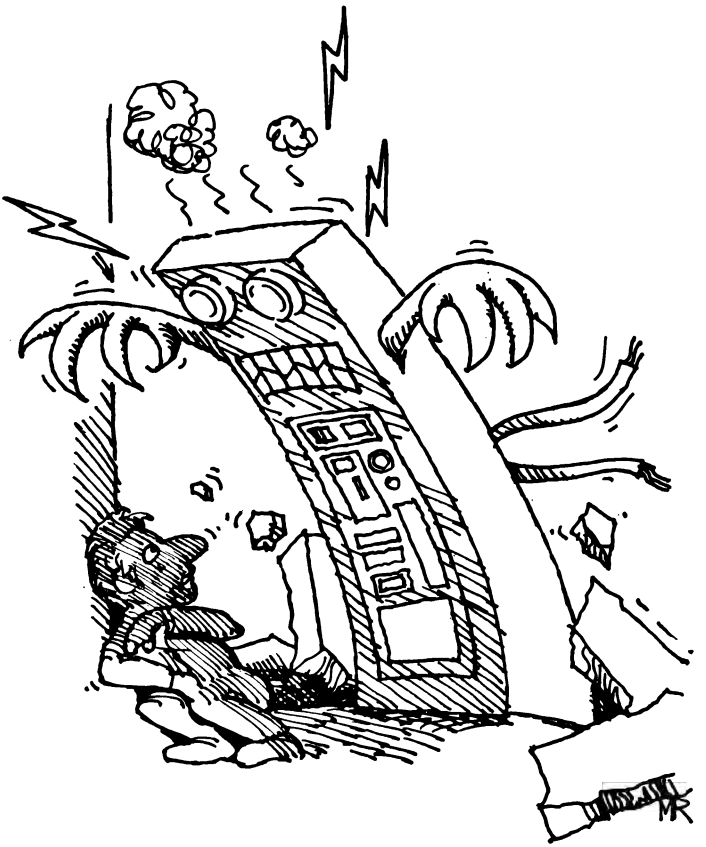
Historically, society has always (in part at least) resisted change, sometimes physically as with the often-quoted Luddites who smashed up textile mills in the 1800s. In some cases the rise of technology has had direct and terrible consequences. One technological advance produced a weapon that caused waves of protest and forecasts of the ending of civilisation. No, it was not the atom bomb; it was the crossbow. The atom bomb came much later, followed by the hydrogen bomb and then the neutron bomb. But these are the more dramatic examples; the steam engine, "horseless carriage" and aeroplane also drew their share of protest. Today the computer is the target.

Most people are afraid or nervous of computers. They cannot understand how computers function and see them as the ultimate threat. Individuals imagine their lives being taken over and run by machines over which they have no control. Knowing that one's name is "on a computer" is enough to instil a deep feeling of insecurity in some, and others feel an intense hatred for the impersonal nature of this method of keeping information.

Yet another group of people – those involved in manufacturing or production industries – sees computer technology as some form of automation, which immediately signifies unemployment. While scientists and engineers have visions of totally automated factories, unions maintain their traditional attitude of confrontation towards the concept.

Generations raised on a television diet of mad scientists and monsters and the day's disasters now served up in living colour complete with sound are also wont to think of computers as taking over the world. It is little wonder that this attitude prevails.

A great deal of this confusion is happening because we are being accelerated into a new phase of development. Sociologists



tell us that our society first passed through an agricultural era and then came the industrial age where society was based on the factory and production line. We are now into what has been called alternatively "the micro-chip revolution", "the silicon-chip era" and "the information explosion" – take your pick.

Whatever its label, there can be absolutely no doubt that tomorrow is not going to be just some sort of streamlined version of today. It is going to be different and the name of the game will be adaptation.

In the past there have been numerous advances in technology which have found ready acceptance – radio and television are two examples. Carbon paper has given way to the plain paper copying machine; and in the home there are automatic stoves, dishwashers, washing and sewing machines and video recorders which perform wonders at the turn of a dial.

Some of these home appliances have existed in a semi-automated form for years, but the development of solid-state electronics, together with miniaturisation, enabled the production of

versions that offered a whole range of improved features, operated more efficiently and took up less space. A high proportion of these labour-saving devices are fitted with some form of computer technology, yet they are considered "safe" by those who fear the computer itself. People feel that they are in control of home appliances because there is no obvious sign of the microprocessor and its keyboard and screen.

Like it or not, the computer is here to stay and it will play an increasing role in day-to-day activities, but we are still a long way from the completely computerised and technological society created by science-fiction writers. Computer technology will have to become as affordable as the disposable plastic ballpoint pen before we reach a situation where there is a computer in almost every home.

Attempts to predict accurately new fields of computer application are probably doomed to failure. With the speed of developments today's discovery could well be next week's stale news. But despite this handicap, enough has been written about and discussed to enable us to make some educated guesses. Changes have already taken place. They can be seen as indicators which may well point to the pattern of future society.

Will increasing technology mean unemployment? Of course it will, but not in its usual historical sense. We are unlikely to see half the nation out of work with the other half doling out portions of instant soup and bread crusts. Instead, people will have to learn to adapt.

Some occupations will become redundant, but others will rise in their place. The advent of the "horseless carriage" eventually displaced the village blacksmith, who then put on a different hat and evolved into the engineer of today. Unlike the coopers of yesterday, today's barrel makers are more likely to be operating an aluminium-extrusion plant.

One of the most noticeable changes brought about by the advent of computer technology is taking place in the business world – it is the "electronic office". Word processors are starting to fill the former roles of typewriters; filing cabinets are giving way to floppy disks and microfiche machines. Videotapes have become an extension of training manuals. Some executives now do their own typing through desk-based terminals, and documents for entire offices are reproduced by a single high-speed printer.

The role of the office secretary is also changing. In more enlightened organisations the secretary now takes a much more active role in day-to-day office affairs than previously, even to the point of being responsible for certain executive decisions.

Inter-city conferences are held over video link-ups and inter-office business is conducted by "electronic mail" from terminal to terminal over public or private networks.

Twice a day all the major cities swarm like antheaps as commuters scurry to and from their workplaces. Commuting in the future will be at a much lower level as offices decentralise. Even the concept of the office is already undergoing a drastic change as some organisations employ workers in their homes and communicate with each other and a central office through company-installed terminals.

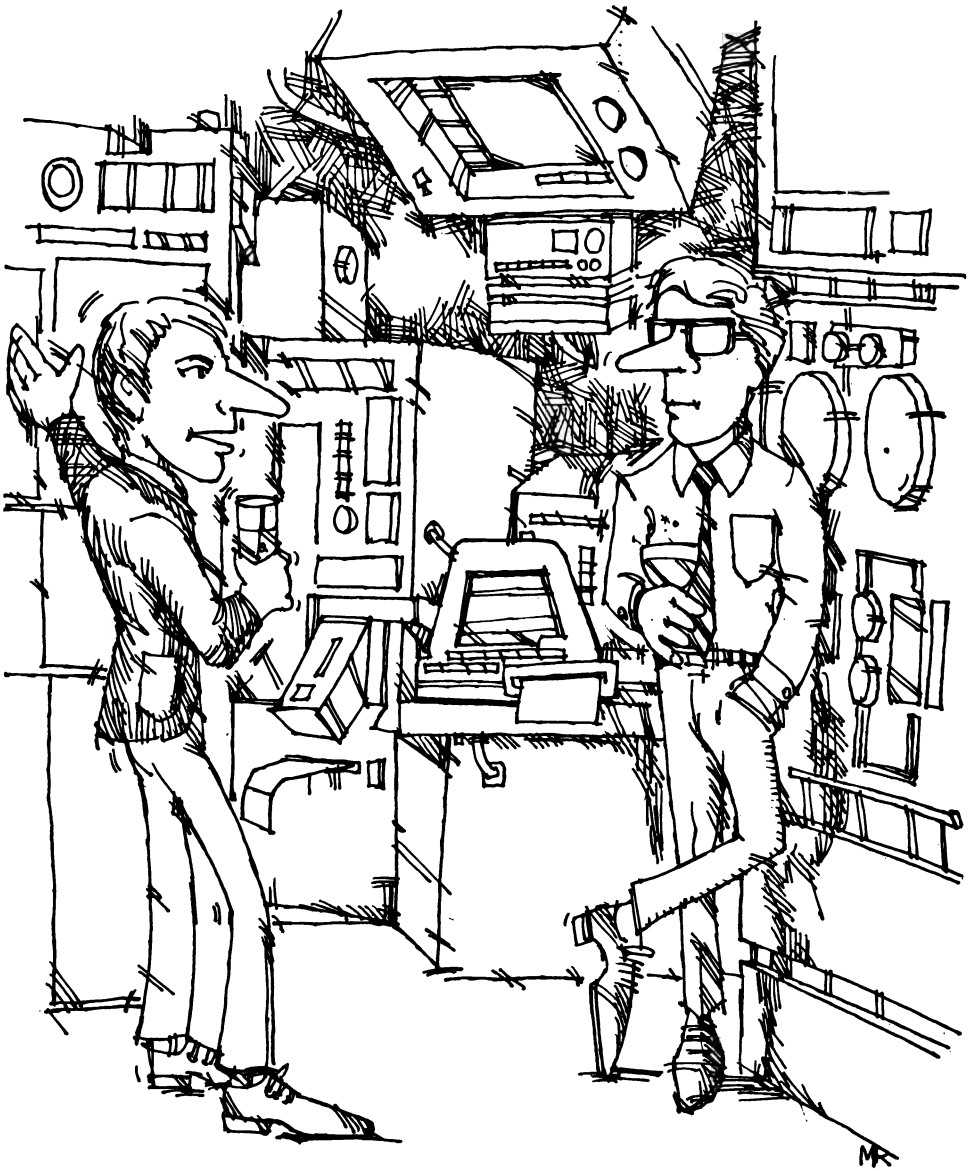
Large radio networks are becoming less important. They are being replaced with local AM or FM stations whose sole purpose is to serve a specific community. Today, in Japan and Canada, there are experimental television stations where viewers can not only participate but also be visually included in studio programmes while remaining in their own homes.

As the working week becomes shorter and shorter, commuters, who today often live some distance from their workplace, will find that it becomes more convenient and less costly to have their working offices at home. In essence many business activities consist of passing pieces of paper to other desks or offices and holding telephone conversations. Apart from the personal contact in an office situation it matters little whether the next office or desk and its telephones or terminals are located in the same building or across the other side of town.

We read of homes where not only the environment but every other function is computer controlled. Currently the price tag is too high for this to be more than the privilege of the wealthy, but in other ways the home of the future will take on new importance as leisure time increaces. Society will accept increased job sharing, even by couples who share the same or complementary occupations, and a single job could indeed become a family concern.

While increased leisure time appeals to most people, there are some who wonder how on earth they will occupy themselves. We will be looking at new industries, and even new forms of recreation, to cater for them.

It is already obvious that the new technology is replacing some of the roles of books, newspapers and other publications. Computer magazines are now available in print or on disk. A high-fashion quarterly magazine has given up paper in favour of videotape. Technical and archival documents and even consumer magazines are available on microfiche cards. Newspapers are involved with various videotext systems.

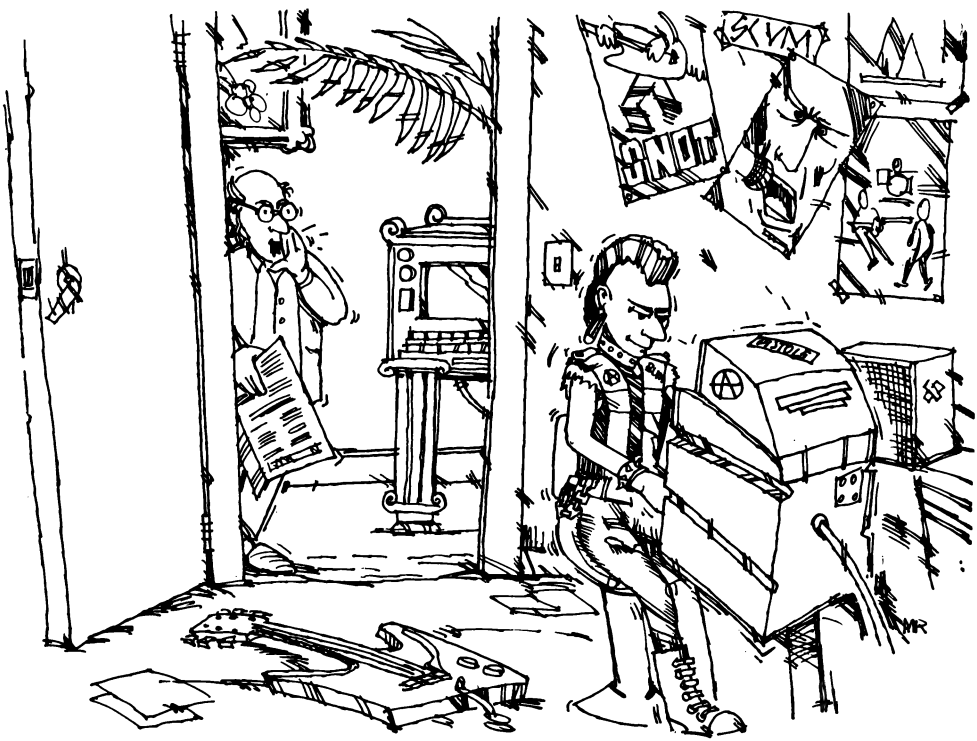


"Whatever happened to the office party?"

Publishing as we know it will indeed change, but the screen will not necessarily replace the printed word. The screen may be a quicker and cheaper means of accessing reference material found in a public database, but it is still the publisher who is responsible for the ultimate form the information takes. Books are so much a part of our culture that they are unlikely to be phased out, but the type of book we buy may well change. In the case of the electronic magazine, a reader is likely to access only those sections that appeal and pay accordingly. Instead of being faced with piles of magazines you don't want to throw away, the same article in the electronic magazine could be accessed time and time again as required.

Behind all these changes is new technology which will continue to improve. We cannot turn the clock back, and the computer is one of the tools that will enable us to adapt.

We speak of the world becoming smaller and smaller. In the days of shipping, a voyage to an overseas country was a fairly leisurely affair. With the introduction of passenger aircraft and



"Turn that computer down!"

then the jet engine weeks of travel were reduced to a few hours. Communications improved, until today we can sit casually in our own homes watching an event on the other side of the world as it happens.

Whatever our reaction to these changes, they have always been rather impersonal and we have exploited the advantages only by choice. But now new technology has caught up with us and probed into our personal lives. Most prominent of all is the computer.

Computer installations have always seemed remote and ominous, largely due to the mystique built around them by computer professionals and later by the public itself. However, this situation is rapidly changing. With the introduction of the personal computer people can now experience computing at first hand. Once again new technology has enabled mass production of something that formerly belonged to a few.

Microcomputers have evolved into a major growth industry in a very short space of time. Models range from the very sophisticated to simpler versions costing under a hundred pounds. Computer shops have opened around the world and schools are now producing a new breed, the 12- and 14-year-olds whose computer expertise leaves their elders scratching their heads in wonderment.

The mass production of computers will have a similar effect on society to the production of the Model T Ford in 1908. The computer will become as common as the transistor radio and will be used as casually as the pocket calculator is today.

Microcomputer users have banded together to form local groups, national clubs and even international associations. Users in widely separated locations can "meet" each other via their computers and the telephone system. With the personally owned microcomputer thousands of people are finding that the world is even smaller and distances are fast becoming irrelevant. The biggest thrill, however, is that technology, once remote, is now available in an easily understood and readily accessible form and participation is on a personal level.

The proliferation of microcomputers is not going to wipe out resentment of new technology. People will always react against any major social change, although future generations may be more ready to adapt. The microcomputer itself is very adaptable. It matters not at all how quickly you input information and whether you use your fingers, feet or even a stick held between your toes to operate the keyboard.

The impact of the microcomputer on the world of the handi-

capped is incalculable. The disabled, with bodies out of action but brains active, have found that microcomputers have enabled them to play a greater role in society. Microcomputer users were quick to find ways in which these new machines could assist. Special groups were formed and software was written to develop this application.

There are speech synthesizers which can "speak" each character, word and sentence as they appear on the screen. This has enabled the blind to "see", and blind people can operate microcomputers with the confidence and expertise of sighted people using conventional typewriters. Quadriplegics can write their own programs and use their computers to communicate with the outside world. Although the world may seem to have grown smaller, for the handicapped the microcomputer has opened up horizons that seemed forever closed to them.

As microcomputers become more familiar and new generations grow up regarding them as everyday appliances, people will be less likely to treat "the computer" as a whipping boy for all their ills. Perhaps they will realise that it's not the machine itself that annoys them, but the bureaucracy it represents.

We are undergoing change and we are heading into a new era. What an exciting time must lie ahead.

Chapter 11

So you want to know more

This book was intended to give a general introduction to microcomputing, but not to pursue the many aspects in any great depth. If you have come this far you will be looking at this chapter either out of sheer curiosity or interest in learning more. If the latter there are several ways open to you.

If you feel you need some direct tuition make enquiries at your local technical institute or polytechnic. Short courses, dealing with the technical make-up of computers or with programming at both elementary and advanced levels are often held during evenings and weekends. Such courses usually involve "hands-on" computer experience. ("Hands-on" means being able to use a computer when and where you want to, such as at your office desk or at home. One of the reasons this term arose was that in the past people were used to having others do the actual operating, or they had to book time on a system with the disadvantage of not being able to continue with a particular task until the computer became available again.)

For those who would like to read more about programming in the BASIC language, any technical or larger bookshop should be able to help you. If stocks are not actually held, they will be able to order books for you.

When ordering by title alone you can never be sure what you are going to get. However the following books all deal with BASIC and should be easily followed by a beginner. When read in conjunction with your computer manual they should help you learn how to program and will offer plenty of ideas and examples.

One of the problems of BASIC, already mentioned, is that of the various dialects. A program written for the TRS-80 model will not run on an APPLE II, and a program for the COMMODORE will not run on a TRS-80. Although the programs themselves can be adapted, the commands for various computer

functions will be different and unless you have a whole range of manuals you will spend a long time trying to work them out.

If you have any intention of rewriting or adapting programs not specifically written for your computer there is one book that you should have above all others. Called *The BASIC Handbook* and written by David Lien, it is recognised as the encyclopaedia of the BASIC language. It explains over 250 BASIC words, how they are used and alternative strategies if your machine doesn't have them. The publisher is Compusoft Publishing, San Diego.

For beginners, the following are recommended:

Illustrating BASIC by Donald Alcock, Cambridge University Press.

BASIC and the Personal Computer by Dwyer & Critchfield, Ohio Scientific Inc., Aurora, Ohio.

Learning Level II by David Lien, Compusoft Publishing. (This book is intended for the TRS-80.)

A Guided Tour of Computer Programming in Basic by Thomas A. Dwyer and Michael S. Kaufman, Harrap, London.

Stimulating Simulations by C. W. Engel, Hayden Book Company Inc., New Jersey.

Computer Games (for business, schools and homes) by J. Victor

Nahigan and William S. Hodges, Winthrop Publishers Inc., Cambridge, Massachusetts.

BASIC Computer Games also *More BASIC Computer Games* edited by David Ahl, Creative Computing Press, P.O. Box 789-M, Morristown, New Jersey 07960.

Instant BASIC by Jerald R. Brown, Dilithium Press, P.O. Box 92, Forest Grove, Oregon 97116.

For additional reading try these:

Advanced BASIC – Applications and Problems by James S. Coan, Hayden.

Discovering BASIC – A Problem Solving Approach by Robert E. Smith, Hayden.

The A to Z Book of Computer Games by Thomas C. McIntire, TAB Books, Blue Ridge Summit, Pennsylvania 17214.

The Most Popular Subroutines in BASIC by Ken Tracton, TAB Books.

Programming in BASIC by Ralph M. Stair Jr, Richard D. Irwin Inc., Homewood, Illinois.

Program Design edited by Blaise W. Liffick, BYTE Publications Inc., 70 Main Street, Petersborough, New Hampshire 03458.

If your interest is more general and you want to keep up with the whole field, there are a number of magazines available which cover the whole computing field as well as those that favour one particular brand of microcomputer.

General Magazines

U.S.A.

Byte, BYTE Publications Inc., 70 Main Street, Petersborough, New Hampshire 03458.

Personal Computing, 4 Disk Drive, Box 13916, Philadelphia, Pennsylvania 19101.

Creative Computing, Creative Computing Press, P.O. Box 789-M, Morristown, New Jersey 07960.

Kilobaud Microcomputing, P.O. Box 997, Farmingdale, New York 11737.

AUSTRALIA

Your Computer, P.O. Box 216, Spit Junction 2088, N.S.W.

Australian Personal Computing, P.O. Box 115, Carlton 3053, Victoria.

Specialised Magazines

TRS-80

80 Microcomputing, P.O. Box 997, Farmingdale, New York 11737.

APPLE II

Nibble, P.O. Box 325, Lincoln, 01773.

Apple Orchard, P.O. Box 1493, Beaverton, Oregon 97075.

COMMODORE

Commodore Magazine, P.O. Box 336, Artarmon 2064, N.S.W., Australia.

Printout, P.O. Box 48, Newbury, Berkshire RG16 0BD, England.

The 6502 Resource Magazine.

Compute, P.O. Box 5406, Greensboro, North Carolina 27401.

(This magazine covers PET, APPLE, ATARI, OSI, KIM, SYM and AIM models.)

This is by no means a comprehensive list of reading, but you will discover others and will soon be able to select for yourself.

Glossary

ACCESS To make use of or gain access to certain programs or information.

ACOUSTIC COUPLER A portable device consisting of two rubber cups which fit over the earpiece and microphone of a telephone receiver. The coupler converts computer information into a series of tones which are transmitted over the telephone to another computer. See also **MODEM**.

ALPHANUMERIC Information containing both letters and figures.

ASSEMBLY LANGUAGE A form of programming where the computer's machine-language instructions are written in a symbolic form. An "assembler" then converts the program to machine language.

ASSEMBLY LANGUAGE PROGRAMMING To write a program using assembly language.

BINARY A mathematical numbering system using ones and zeros only.

BASIC A high-level computing language composed of English terms.

BUG Any problem or irregularity that occurs within a computer program.

BUS A group of wires or printed circuitry which enables various parts of a computer to exchange information.

BYTE A term of measurement for a unit of information, roughly equivalent to a letter or space on a printed page.

COMPUTERESE or **COMPUTERSPEAK** Jargon used by computer specialists.

CPU Central Processing Unit – the "heart" of the computer.

DAISY-WHEEL A printer capable of producing high-quality printing using conventional typefaces. The type is arranged on a wheel at the end of long fingers, which are similar to the spokes on a bicycle wheel or the petals of a daisy.

DATA The raw material used to provide information.

DATABASE A program which stores processed data in files for future reference.

DIGITISED INFORMATION Information broken down into binary-coded form.

DISK DRIVE A mechanical device used for operating diskettes.

DISKETTE Also known as a "disk" and "floppy disk". A flexible disk covered with magnetic oxide and sealed in a protective sleeve. It is used for the storage and retrieval of computer programs.

DOT-MATRIX A printer which forms its impressions by a series of fine wires striking through a ribbon. Letters and so on are produced as a series of closely placed dots.

ERROR TRAP A routine built into a program to detect errors entered through the keyboard by a user.

FIRMWARE Programs residing in ROM which cannot be accessed by a user.

FLOPPY DISK See diskette.

FLOWCHART A diagram or chart which represents the flow of data, methods, steps and procedures, etc., followed for solving a specific problem.

FORMAT, FORMATTING To arrange data to a predetermined plan, layout or style.

FRICTION FEED Where paper is held between a roller and platen and advanced by gear-driven rollers, as with a conventional typewriter.

FUNCTION A sub-program that processes variables in some well-defined way.

HARD COPY A term usually meaning output from a printer.

HARD DISK Similar to a floppy disk in purpose, but manufactured on a non-flexible metal base and usually sealed within its own drive system. Capable of storing large amounts of information.

HARDWARE The mechanical and electronic components of a computer system.

HEXADECIMAL A method of expressing numbers in machine-language programming.

HIGH-LEVEL LANGUAGE A computer-programming language which employs a precise form of English, enabling programs to be written by non-specialist computer users.

INFORMATION Knowledge that is communicated or received.

INPUT Any information passed to the computer.

INTEGRATED CIRCUIT A self-contained miniature electronic circuit. In a microcomputer this is a flat, rectangular plug-in device with metal pins along each side.

INTERFACE To connect some device to the computer so that both components are compatible.

INTERPRETER The means whereby a program is translated by the computer from a high-level language to machine language.

KEYBOARD Same as a typewriter keyboard and used for passing information to the computer.

KILOBYTE 1,024 bytes (often referred to in round figures as 1,000 bytes). Also expressed as the letter K.

LAYOUT An arrangement of data or information according to a plan.

LINE NUMBER Computer programs in BASIC are written line-by-line, each being identified by a number.

LOW-LEVEL LANGUAGE A term used for assembly and machine languages.

MACHINE LANGUAGE or **ASSEMBLY LANGUAGE** A method of programming where the programs are written in binary code.

MAINFRAME A very large computer or computer installation.

MARK-SENSE CARD A pre-printed card on which information is noted by pencil or pen mark and then optically scanned by a card reader, which passes the selected information to the computer.

MEMORY Usually the amount of RAM available to the user.

MICROCOMPUTER A small computer, self contained, portable and operated by a microprocessor.

MICROPROCESSOR A group of electrical circuits which performs the essential functions of the Central Processing Unit.

MICROSECOND One millionth of a second.

MILLISECOND One thousandth of a second.

MINI-DISK Diskettes come in two sizes, the smaller sometimes referred to as a mini-disk.

MINICOMPUTER This term is usually applied to computers of a size and capability between a microcomputer and mainframe system. Some also use a mainframe for storage of information.

MODEM A device permanently wired into the telephone system used for communicating with another computer in a different location. See also **ACOUSTIC COUPLER**.

MONITOR A term used for a VDU.

NANOSECOND One thousandth-millionth of a second.

OPERATING PROGRAM A set of instructions stored in ROM, which tells the computer what to do when it is switched on, which key has been pressed and so on.

OPTICAL MARK-SENSE CARD READER An optical device which reads information from mark-sense cards and transfers it to the computer.

OUTPUT Any information generated by the computer, displayed on the VDU or passed to an external device such as a printer.

PERIPHERALS Various pieces of equipment connected to the computer, such as a disk drive or printer.

PERSONAL COMPUTER A microcomputer which is a self-contained system and is used as required by one person.

PINFEED A system for feeding paper through a printer. Metal pins mounted around the roller fit through holes along the margins of the paper and keep it perfectly aligned. Pinfeed systems are usually not adjustable.

PRINT In BASIC programs this term can be used either to generate a line of output or to insert a blank line.

PRINT-OUT Hard copy from a printer.

PROGRAM A set of instructions written in a language understood by the computer.

PUNCH CARDS Cards containing holes punched in a particular sequence as coded information. These are passed through a reader which transfers the information to the computer.

RAM Random-Access-Memory: the type of memory provided as workspace for the user where information can be changed or processed at will.

READ To retrieve information (for instance, from a disk) and transfer it into the computer.

ROM Read-Only-Memory: the type of memory usually containing the computer's operating language and instructions. The user cannot gain access to the contents of ROM.

SECTOR A subdivision of a track on a disk.

SEMICONDUCTOR A solid electrical device used to make diodes, transistors and integrated circuits.

SILICON CHIP A small piece of silicon on which a large number of electrical circuits have been reproduced.

SOFTWARE The internal programs of a computer, or routines prepared for the computer as programs.

STATEMENT A written instruction within a program.

STRING VARIABLE A variable used to represent a "string", or piece of information within a program containing either letters or both letters and numerals.

SUBROUTINE A short routine or mini-program within a computer program, usually with some recurring function.

SYSTEM MONITOR A term sometimes used for the set of instructions stored in ROM that make up the operating program for the computer.

TERMINAL A VDU and keyboard remotely connected to a central computer.

TEXT FILE A file of information generated by a computer program or data file.

THERMAL PRINTER A printer which uses an electrical contact to burn an image on heat-sensitive paper.

TRACK A division on a disk for the storage of information, similar to the tracks on a musical record.

TRACTOR FEED Similar to pinfeed, but with adjustable sprockets. The sprockets are driven so the paper is pulled through the printer one line at a time.

TRANSFER RESISTOR A term first applied to the transistor.

VARIABLE A symbol used in programming to represent a specific piece or amount of data.

VDU Visual Display Unit. A screen on which information from the computer is displayed, similar to a television screen.

WRITE To record or store computer information on a disk.

INDEX

abacus 3
access 105
accessories 12
acoustic coupler 35, 105
Aiken, Howard 8
ALGOL, description of 51
alphanumeric 9, 105
Analytical Engine 6
appending text files 75
apple II 72
applications, business 37
applications, educational 41
applications, farming 39
applications, home use 42
applications, recreational 44
applications, scientific/technical 42
assembly language 23, 49, 105, 107
assembly language programming 105
ATLAS 10

Babbage, Charles 6
BASIC 20, 105
BASIC, description 50
BASIC, dialects of 50
Bell Telephone Company 9
binary 105
binary system 22
bug 105
bus 105
buyer's checklist 91
byte 105
byte, definition 14

card reader 18
cassette recorder 12
cassette, program storage 14, 30
Central Processing Unit 13
choosing a microcomputer 82
choosing a system 84
clay tablets 2
COBOL, description 51
COLOSSUS 8
commercial production 9
commodore 14
communications 35

computer-controlled operations 28
computer courses 101
computer kitsets 10
computer languages 47
computer music 36
computer speech 36
computer users, classification 20
computer, do you need it? 82
computerese 11, 105
computers and our future 92
computers and the handicapped 99
computers in the home 99
computers, infallible oracles 83
computerspeak 11, 105
CPU 13, 105
CPU types 13
Currency Converter 73
custom software 91

daisy wheel 106
Dartmouth College 50

data 106
data, definition 17
database 92, 106
Difference Engine 6
digital storage 14
digitised information 30, 106
digitisers 36
disk drive 12, 15, 106
disk drive, description 31
disk operating system (DOS) 15
disk sectors 31
disk storage 32
disk tracks 31
disk, diskette 15
diskette 106
disks, disadvantages 31
dot matrix 106

Eckert, Prosper 8
EDVAC 9
ENIAC 8
error trap 106

firmware 17, 106
floppy disk 15, 106
flowchart 106
flowchart symbols 64
flowcharts 64–68
format, formatting 106
FORTH, description of 52
FORTRAN, description of 51
friction feed 106
function 106

game paddles 36
Gunter, Edmund 4
Gutenberg, Johann 3

hands-on, 101
hard copy 15, 106
hard disk 15, 32, 106
hardware 12, 106
Harvard Mark 1 8
Harvard University 8
hexadecimal 24, 47, 107
high level language 22, 107
high level languages 49
Hollerith, Herman 7

IBM 7
information 1, 107
input 17, 107
Integrated Circuit 107
Integrated Circuit (IC) 16
interface 28, 107
interpreter 23, 107

Jacquard, Joseph-Marie 7
jargon 11
joysticks 36

Kemeny, John 50
keyboard 12, 14, 107
kilobyte 107
Kurtz, Thomas 50

layout 107
Leibnitz, Gottfried 6
line number 107
low level language 107
low level languages 49

machine language 23, 49, 107
magnetic tape 8
mail order software 90
mainframe 107
mainframe computers 18
making the commitment 84
mark sense card 107
mass production 10
Mauchly, John 8
memory 107
memory addresses 26
memory, explanation 24
memory, size 13
microcomputer 10, 12, 107
microcomputers 19
microprocessor 12, 107
microsecond 17, 107
millisecond 17, 107
Milne, Henry 3
minidisk 108
minicomputer 10, 108
minicomputers 19
MODEM 35, 108
monitor 14, 108
Moore, Chares 52

nanosecond 17, 108
"Napier's Bones" 4
Napier, John 4

operating program 108
optical mark sense card 18
optical mark sense card reader 35, 108
output 17, 108

paper 2

papyrus 2
parchment 2
Pascale, Blaise 4
Pascal, description of 51
peripherals 29, 108
personal computer 108
PILOT, description of 52
pinfeed 108
PL/1, description of 52
plotters 34
PRINT 26, 108
print out 108
print-out 15
printer, daisy wheel 33
printer, dot matrix 32
printer, friction feed 34
printer, pin feed 34
printer, thermal 32
printer, tractor feed 34
printer types and uses 34
program 108
program checking 70
program coding 58
program design 55
program guidelines 63
program listings 71
program writing 24, 55–63
program, definition 13
programming, approaching it 53
publishing 96

punch cards 7, 18, 108

radio valves 8
RAM 16, 108
Random Access Memory 16
read 108
Read Only Memory 16
read, definition 31
recommended reading 102
reliability and faults 87
Remington & Sons 7
Remington, E. 3
ROM 16, 108

Salary Analysis 72
screen 12, 14
sector 109
semiconductor 9, 109
Sholes, Christopher 3
silicon chip 13, 109
slide rule 4
social resistance 93
software 13, 109
software selection 89
statement 109
string variable 27, 109
subroutine 109
subroutines 69
system monitor 16, 109

Tabulating Machine Company 7
technical software selection 89
technology acceptance 94
temperature monitoring 28
terminal 109
text file 109
the electronic office 95
the information explosion 94
The Name Game 78
thermal printer 109
track 109
tractor feed 109
transfer resistor 109
transistor 9, 12
transistor development 9
TRS-80 76
typewriter 3

U.S. Census Bureau 7
unemployment 95
UNIVAC 9
University of Pennsylvania 8

Vacuum tubes 8
variable 109
variables 25
VDU 14, 109
video interface 36
Visual Display Unit 14
voice recognition 36
von Neumann, John 9

wartime development 8
Wirth, Niklaus 51
write 109
write, definition 31
writing instruments 3

ANYONE CAN USE A MICROCOMPUTER. YOU DO NOT HAVE TO BE TECHNICALLY MINDED. That is the message of this book.

★ Find out what a microcomputer can do for you
- in the business world
- in farming and horticulture
- in the laboratory
- in education
- in industry
- in the home and for recreation

★ Learn how a microcomputer works

★ Banish forever any fears of jargon – this book puts you in the know

★ Find out how to choose the right equipment and programs to suit your needs

This is the computer book you should read first before all others. Written in a clear, precise and often humorous style, it *assumes no previous knowledge* of computing and offers a painless initiation into the microcomputer revolution.

"This book is one of the very few New Zealand-written works on this crucial subject and, as such, deserves to be a fixture on the bookshelves of public libraries, universities, at work and in the home."

Peter Isaac,
Computer columnist, The Dominion

A REED BOOK ISBN 0 589 01466 8

Book Tokens
Give them the pleasure of choosing
Book Tokens can be bought and exchanged at most bookshops

Microcomputers in Plain English *for New Zealanders*

Brian D. Strong